[Proc. SPIE Vol. 1445, "Medical Imaging V", pp. 297–311, San Jose CA, February 1991]

Efficient computation of various types of skeletons

Luc Vincent

Harvard University, Division of Applied Sciences
Pierce Hall, Cambridge MA 02138, USA

Abstract

The skeleton transformation is particularly useful in the field of image processing and may be computed by various techniques. This paper describes a new skeleton algorithm, which is based on the original concept of *anchor points,* i.e. of points that the skeleton is bound to contain. Its first step consists in extracting the desired anchor points. Then, the set X to skeletonize is progressively thinned in such a way that, on the one hand, the anchor points can never be removed and on the other hand, the homotopy of X cannot be modified. This operation is efficiently implemented thanks to a queue of pixels. The algorithm turns out to be extremely efficient and accurate on conventional computers. Furthermore, it allows to deal not only with standard skeletons, but also with such objects as *minimal skeletons, homotopic markings,* smoothed and pruned skeletons, etc. Its flexibility is also proved by the facts that it works both in the Euclidean and the geodesic cases, that its adaptation to any kind of grid is straightforward and that it can even be extended to n-dimensional images and to graphs! It is expected to contribute to new insights into the use of skeletons in image analysis and shape recognition.

1 Introduction

Skeletonization is a widely used transformation in the fields of image analysis and shape recognition. The concept of skeleton was introduced by Blum in 1961 [4], under the name of *medial axis transform.* The definition proposed by Blum corresponds to the intuitive skeleton notion, a sort of minimal representation of a set X under the form of lines of unit width. It is based on the concept of prairie fire: assuming a fire is propagating within X at a uniform speed starting from its contours, the medial axis of X is the set of the points where different firefronts meet (see Fig. 1). A more formal definition of skeletons in both the discrete and the continuous cases* was proposed by Calabi in 1966 [6] and is based on the concept of *maximal ball.* This definition tends to be more widely used in the field of morphological image analysis [17], and this is the reason why we recall it in § 1.1. In this paper, we consider both approaches.

The topological study of the skeleton in the continuous plane $\mathbb{R}^2$ turns out to be extremely complex. One of the major results states that the topological closure of the skeleton of a connected open set is itself connected [10, 11]... Similarly, in the discrete case, the skeleton notion is far from being a trivial one. Indeed, as we shall see later, the direct application of Calabi's definition leads to skeletons which do not preserve the homotopy of the sets on which they act. Various algorithms have been proposed to remedy this inconvenience, none of which is completely satisfying: they either produce skeletons with parasitic dendrites, or are computationally inefficient, or only allow the computation of *one* given skeleton-like transformation. In fact, the literature on skeletons is so abundant that it becomes impossible to review all the existing works exhaustively. We shall just briefly mention below the main categories of skeleton algorithms, so as to show in what respects the present approach is novel.

*In the continuous case, there exist in fact slight differences between the skeleton by maximal balls and the medial axis transform proposed by Blum.

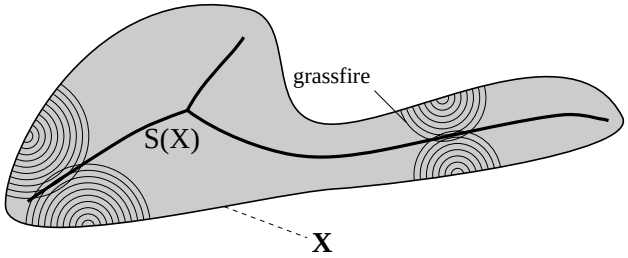


Figure 1 : *The skeleton can be defined as the set of the pixels where two firefronts stemming from the objects boundaries meet.*

In § 1.1, some notations and essential definitions are recalled, while § 1.2 is concerned with the review of the three major families of skeleton algorithms. § 2 is then devoted to the detailed description of the present algorithm, which is provided in a pseudo C language. Its advantages and computational efficiency are highlighted. Additionally, by using different sets of anchor points as input in this algorithm, one can produce a whole range of different skeletons; some of the most interesting among them are reviewed in § 3. In the conclusion, some ideas on the possible extensions of this work are given.

1.1 Notations, Definitions

From now on, we are only concerned with the discrete plane $\mathbb{Z}^2$, equipped with a grid G providing the neighborhood relationships between pixels. The set of the neighbors of a given pixel p according to G is denoted $N_G(p)$. An image I is a mapping from a domain $D_I \subset \mathbb{Z}^2$ into $\mathbb{Z}$. It is binary if it may only take values 0 or 1, in which case it is assimilated with the set $X = \{p \in \mathbb{Z}^2 \mid I(p) = 1\}$. Although the proposed algorithm works in any kind of grid, we chose here to use the hexagonal one because of its good behavior with respect to connectivity [19, chapter 1]. This grid is illustrated by Fig 2. Its elementary vectors are denoted $\vec{u}_0$, $\vec{u}_1$,..., $\vec{u}_5$, and are illustrated by Fig. 3. Accordingly, the 6 neighbors of a pixel p are $p + \vec{u}_0$, $p + \vec{u}_1$,..., $p + \vec{u}_5$.

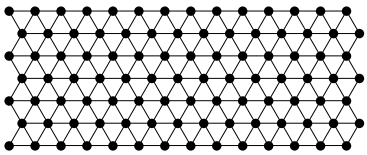


Figure 2 : *Portion of the hexagonal grid.*

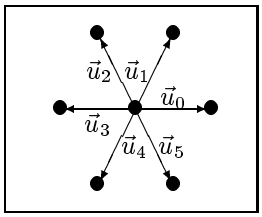


Figure 3 : *The 6 elementary vectors of the hexagonal grid.*

Calabi's definition of the skeleton is based on the following notion of maximal ball:

Definition 1 *A ball B included in X is said to be maximal if and only if there exits no other ball included in X and containing B:*

$$\forall B' \text{ ball}, \qquad B \subseteq B' \subseteq X \Longrightarrow B' = B. \tag{1}$$

This concept is illustrated by Fig. 4. It is worth noting that the balls used throughout this paper are *hexagons*, i.e., the balls of the hexagonal distance (see [19, chapter 1]). The definition of the skeleton follows:

Definition 2 (skeleton by maximal balls) *The skeleton S(X) of a set* $X \subset \mathbb{Z}^2$ *is the set of the centers of its maximal balls:*

$$S(X) = \{p \in X, \exists r \geq 0, B(p,r) \text{ maximal ball of } X\}. \quad (2)$$

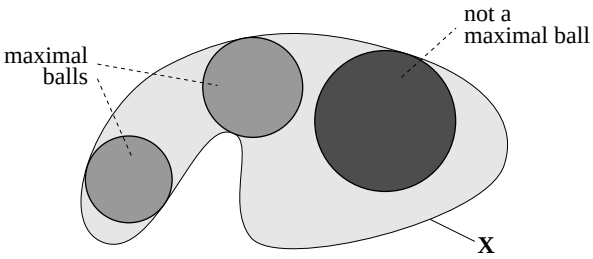


Figure 4 : Concept of maximal ball.

Unfortunately, the skeletons stemming from this definition are often referred to as *non connected skeletons.* Indeed, as shown in Fig. 5, the direct application of definition 2 yields very disconnected skeletons! This example uses the hexagonal grid, but similar difficulties are encountered with square grids in 4- or 8-connectivity...

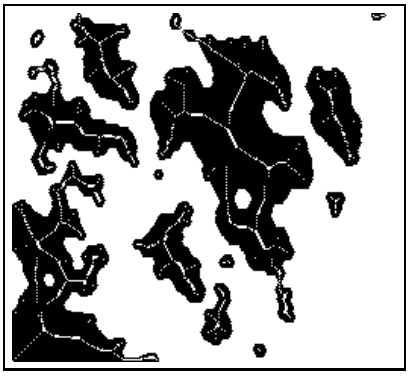


Figure 5 : Skeleton by maximal balls of a set: not connected!

Therefore, the skeleton by maximal balls is of limited practical interest by itself. As quoted in [3], it may sometimes be useful for compressing binary images, since it can be proved that a set $X \subset \mathbb{Z}^2$ is equal to the union of its maximal balls. Additionally, there exist a remarkable property about this "non-connected skeleton". Recalling that the distance function dist_X of a set $X \in \mathbb{Z}^2$ associates with each point of X its distance to the background X^C [15, 5], we can state:

Proposition 3 *The skeleton by maximal balls of X is the set of the local maxima of the distance function, i.e:*

$$S(X) = \{p \in X, \forall p' \in \ N_G(p), dist_X(p) \geq dist_X(p')\}. \quad (3)$$

This equation is typically digital and eludes the intuition of the continuous space. It proves that the skeleton by maximal balls is easily derived from the distance function, a property which shall be used in the following.

The non-preservation of homotopy makes the skeleton by maximal balls definitely useless as shape descriptor: the measurement of its number of extremities, its number of triple points, or even its number of pixels does not provide any interesting information. It is thus essential to obtain a *connected* skeleton, i.e., a homotopic one. To summarize, the algorithmic problems raised by skeletons may be stated as follows:

How can we determine in the most efficient way a set of pixels that it "as close as possible" to the skeleton by maximal balls while preserving the homotopy of the original set?

1.2 Quick review of the existing algorithms

To answer the above question, a considerable number of skeleton algorithms have been published, which cannot be systematically reviewed here. We shall just attempt to classify them into three main families, give some hints on their principles and indicate references on the topic. We do not consider below the skeleton algorithms coming from computational geometry, since they are usually not suited to concrete image analysis tasks.

Homotopic thinnings Historically, the first algorithmic approaches to the computation of skeletons used homotopic thinnings. Their principle consists in removing successive "peels" of the sets to be skeletonized until the one-pixel thickness is reached. At each step, only boundary pixels having particular neighborhood configurations may be removed. The configurations are chosen such that the homotopy of the set is not affected by any pixel removal. In the field of mathematical morphology [17], these configurations are often referenced by means of structuring elements and the peeling operation is called *morphological thinning* [3, 21]. On the hexagonal grid, the most suitable element is L (see Golay's alphabet [7]), which is illustrated by Fig. 6.

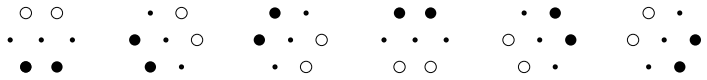


Figure 6 : *The two-phase structuring element L and its 6 rotations.*

These algorithms work *in parallel* over all the image pixels. They are thus particularly suited to hardware implementations and remain widely used in practice. However, the results they produce have in general not much in common with the actual skeleton by maximal balls: indeed, many parasitic dendrites and undesirable deviations tend to appear. Different skeletons may also be obtained depending on the order in which the structuring elements are used... In addition, the execution time of these parallel algorithms is prohibitive on conventional computers, since hundreds of scannings of the entire image are often required!

Crest-Lines of distance functions This second class of algorithms is based upon the fact that the skeleton by maximal balls can be obtained as set of the local maxima of the distance function. These local maxima are in fact a subset of the *crest-points* described by F. Meyer [12]. The idea is to start from these pixels and connect them at best by using the crest-lines of the distance function in *upstream generation* procedures [12, 13]. This latter step can be implemented sequentially [14], so that these algorithms are much faster than those based on parallel thinnings. They are also designed to produce a homotopic superset of the skeleton by maximal balls and are therefore more accurate than the previous ones. However, they imply complex studies of the neighboorhood configurations corresponding to crest points and pixels liable to be propagated during the upstream-climbing, Consequently they cannot be easily transposed from one grid to another; furthermore, as opposed to the algorithm proposed in this paper, they only permit the computation of a small variety of skeletons.

Algorithms based on contours Several authors have tried to directly transcribe in algorithmic terms the description of the skeleton as the locus of the pixels where two firefronts coming from the object's boundaries meet. Among the most interesting methods, we can mention that of Arcelli [1] who propagates the fire inside the considered set with the constraint that no pixel of the medial axis can burn. The algorithms introduced in the present paper also make use of this idea. Other methods for fire propagation use loop techniques [16] or successive contour trackings [23]. These techniques spare the computation time required by continuous scannings of pixel neighborhoods, since the skeleton pixels are only detected from the loops, without having to go back to the image. This makes these algorithms extremely fast, but in return, we face the inconvenience common to most algorithms of this kind: it is

very dificult to transpose them from one grid to another, and it is impossible to extend them to 3-d spaces. Moreover, they will not allow the computation of many different skeletons.

2 Proposed Algorithm

2.1 General Description

The algorithm introduced now relies on the respect of the two following principles:

- *the skeleton we want to determine* must *contain the skeleton by maximal balls,*
- *it should preserve the homotopy of the initial set.*

In order to assure the preservation of the homotopy of the initial set X, successive homotopic peelings starting form the contours of X are performed. Moreover, we also impose that during this peeling, *no pixel of the skeleton by maximal balls can be removed.* More precisely, at a given step, a pixel p may be removed (i.e., given value 0) if and only if the following two conditions are fulfilled:

1. p does not belong to the skeleton by maximal balls $S(X)$,
2. the removal of p would not modify the homotopy.

The propagation inside X starting from its contour pixels is achieved by regarding X as a graph whose nodes are its pixels and whose edges are provided by the (hexagonal) grid. Then, a *queue of pixels* is used to perform *breadth-first scannings* of this graph. This idea has already proved to be particularly interesting in image analysis [18] and more particularly, for the computation of numerous morphological transforms [19, 20]. A queue is a First-In-First-Out data structure: the pixels that are first put into it are those that can first be extracted. In other words, each new pixel included in the queue is put on one side whereas a pixel being removed is taken from the other side (see Fig. 7). In practice a queue is simply a large enough array of pointers to pixels, on which three operations may be performed:

- *fifo_add(p): puts the (pointer to) pixel p into the queue.*
- *fifo_first(): returns the (pointer to) pixel which is at the beginning of the queue, and removes it.*
- *fifo_empty(): returns* true *if the queue is empty and* false *otherwise.*

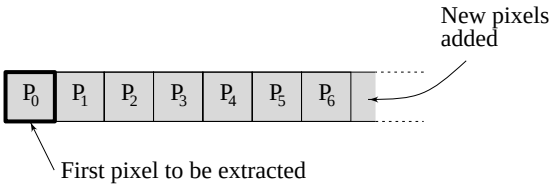


Figure 7: How a queue of pixels works.

After the peeling has been performed until stability is reached, i.e., until the queue is empty, we obtain a skeleton which contains the skeleton by maximal balls $S(X)$: *connecting arcs* of unit width have been added between some components of $S(X)$, yielding a homotopic skeleton. The pixels of $S(X)$ therefore play the role of **anchor points:** the firefronts coming from the boundaries of X tend to anchor themselves on these particular points.

This algorithm lies somewhere in-between Blum and Calabi definitions: Blum's firefront propagation—modelled as a breadth-first propagation in a graph—is contrained by Calabi's centers of maximal balls—provided by the local

maxima of the distance function. Thus, we take advantage of some of the most interesting ideas of the algorithms mentioned in § 1.2: particular configurations of the distance function are used, as in [1] or [12], and successive peelings of the set are realized, as in [16, 23]. In addition, the concept of anchor point turns out to be extremely powerful in practice, since virtually any set can be used as set of anchor points (see § 3).

2.2 Detail of the algorithm

As mentioned above, once the anchor points have been determined, we have to proceed to a homotopic peeling step, in the course of which layers of pixels corresponding to the successive erosions of the set are progressively removed. To rapidly test whether the current pixel should be removed or not, we make use of a look-up table `tab_homo`, determined only once for a given grid type. For example, in the hexagonal grid, there are 2^6 possible configurations around a pixel p, and the index associated with one of them is:

$$\texttt{index} = \sum_{i=0}^{i=5} 2^i \times I(p + \vec{u}_i), \tag{4}$$

I being the binary image to skeletonize. If `tab_homo[index]` $= 1$, the configuration corresponds to a case where p cannot be suppressed without homotopy modification. The look-up table `tab_homo` is shown in Fig. 8 for the hexagonal grid case.

It should be noticed that in the previous paragraph, we have assumed that it is possible to check locally if the removal of a pixel can be done homotopically. In fact, this is not true for general graphs, as explained in [19, chapter 7]. Furthermore, in square grids, if we want a 4-connected skeleton, we cannot decide whether a pixel p can be removed or not without considering its *8-neighborhood!*

Now, if the currently considered pixel, say p, is such that `tab_homo[index]` $= 1$, i.e, this pixel cannot be suppressed, this does not mean that at a later step, p will not become suppressible! Indeed, some of the neighbors of p may have been removed themselves, thus modifying the neighborhood configuration around this pixel[†]. The homotopy around p must therefore be checked again later. One solution is to put p back into the queue immediately. The algorithm then stops when the content the queue is constant over two successive scannings. In fact, doing so would have the disadvantage of considerably slowing down the final part of the algorithm, since the queue is entirely scanned many times to only remove a few pixels at each step!

Therefore, it is much more advisable to store in an array all the (pointers to) pixels which cannot be suppressed immediately. This array is hereafter denoted `TAB`. Once stability has been reached, `TAB` is scanned, and its pixels which are not anchor points and correspond to extremities of branches are again put into the queue. The extremities of branches are detected thanks to another look-up table, denoted `tab_prune`, which takes value 1 for the index of the considered pixel if it corresponds to an extremity, and 0 otherwise (see [19, page 166]). A pruning is then performed, in which no anchor point may be suppressed. At the conclusion of this step, the desired skeleton is obtained.

The complete algorithm is given below in a pseudo-code. For consistency with the rest of the paper, we assume that the anchor image is already available (easily determined as local maxima of the distance function):

Algorithm: Skeleton via anchor points and queue of pixels

- `- input:` $\begin{cases} I, & \texttt{binary image;} \\ I_a & \texttt{binary image of the anchor points;} \end{cases}$
  `- output:` I `/* The skeleton is determined in the input image` I `*/ ;`
- `Initialization:`
  - `tab_homo, tab_prune arrays of size 64;`
  - `TAB: array of (pointers to) pixels;`
  - `index, i: integer variables;`

[†]This can actually never happen with the standard skeleton, see [19, chapter 7].

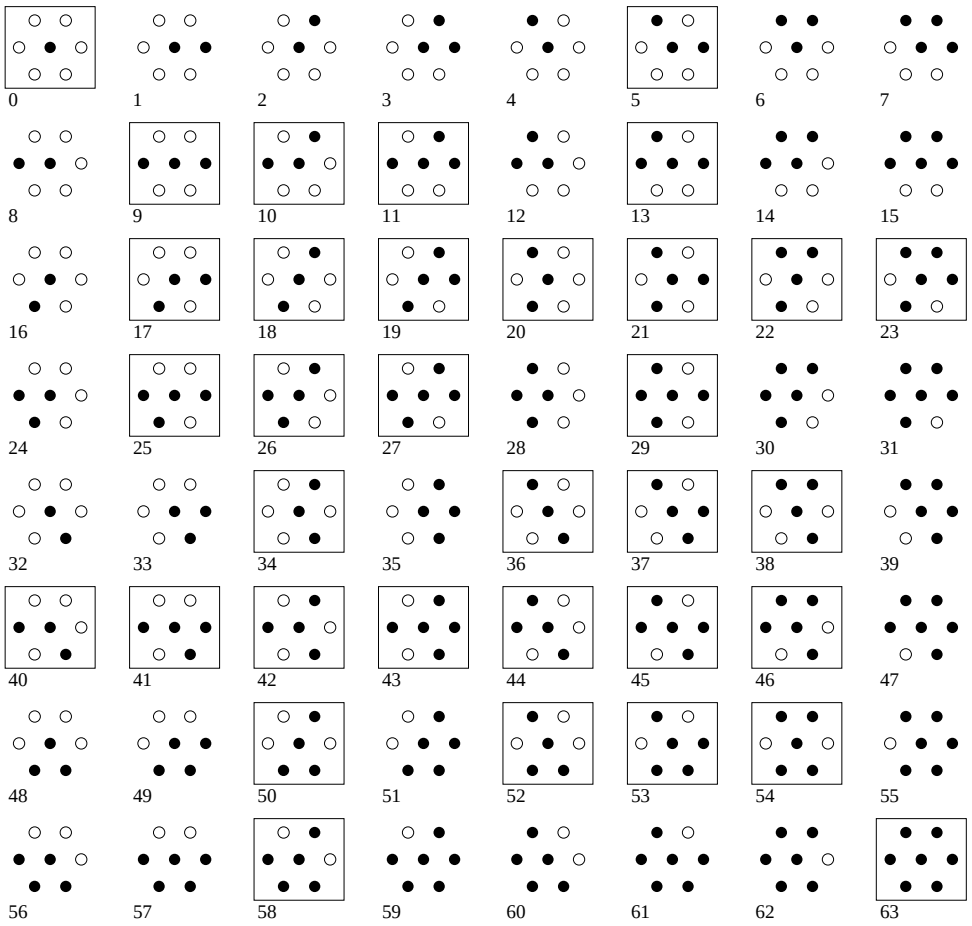


Figure 8 : *The 64 neighborhood configurations around a pixel of the hexagonal grid. Each of them is associated a unique index. Here, the framed configurations have value 1 in array* `tab_homo`*, i.e., correspond to pixels which cannot be removed without homotopy modification.*

- Initialization of the values in I:
  For every pixel p {
    If $I_a(p) = 1$ then $I(p) \leftarrow 1$;
    else if $I(p) = 1$ then $I(p) \leftarrow 2$;
  }
- Initialization of the FIFO structure: border points of origin image in queue
  For every pixel p {
    If $I(p) = 2$ and there exists $q \in N_G(p), I(q) = 0$ {
      fifo_add(p); $I(p) \leftarrow 3$; /* 3 = value of the pixels currently in the queue */
    }
  }

- Propagation:
  While *fifo_empty*() = *false* {
    $p \leftarrow$ *fifo_first*();
    index $\leftarrow 0$; /* Encoding of the neighborhood configuration of p: */
    For $i = 0$ to 5 {
      If $I(p + \vec{u}_i) \neq 0$ then index $\leftarrow$ index $+ 2^i$;
      /* Put the neighbors of p in the queue */
      If $I(p + \vec{u}_i) = 2$ {
        fifo_add$(p + \vec{u}_i)$; $I(p + \vec{u}_i) \leftarrow 3$;
      }
    }
    /* p is put into TAB if it is not suppressible */
    If tab_homo[index] $= 1$ then put p into TAB;
    else $p \leftarrow 0$; /* suppression of p */
    }
  }

- Pruning:

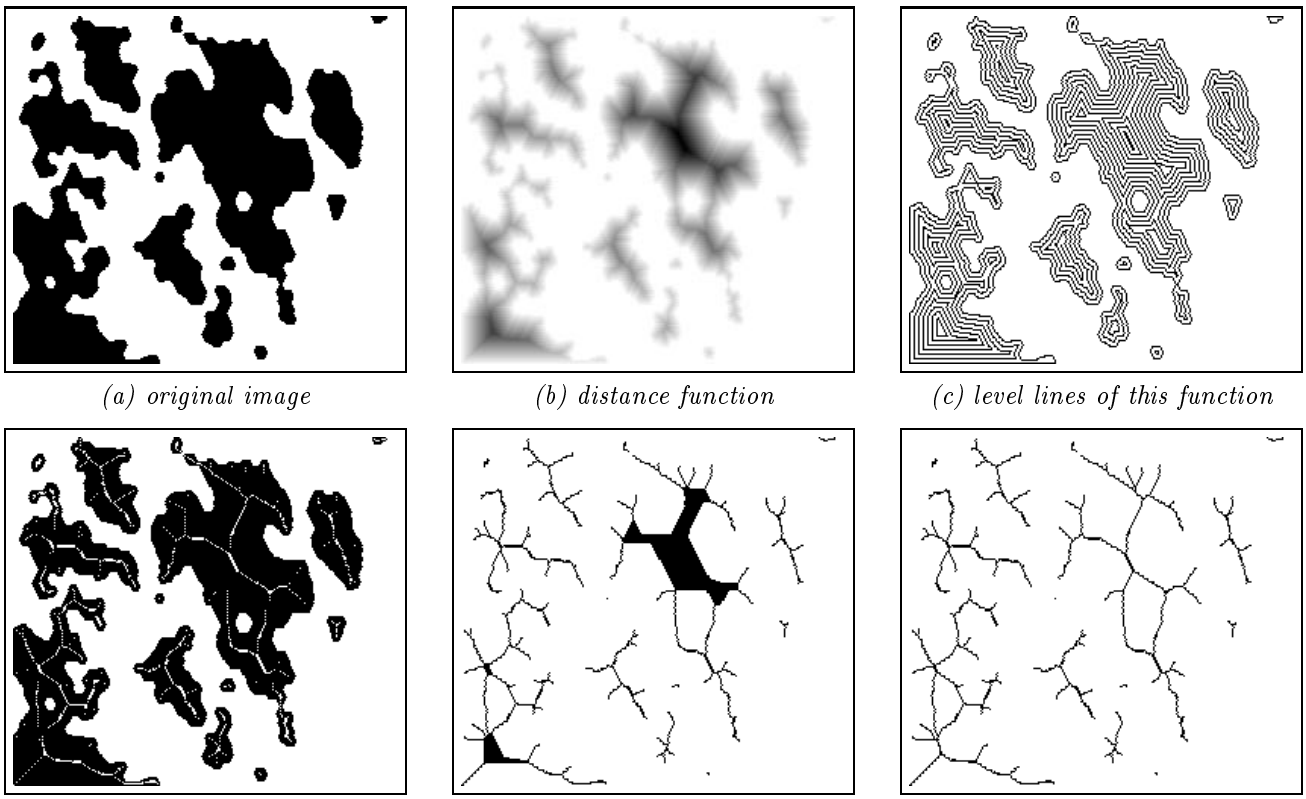
  - Detection of the extremities of branches:
    For every pixel p in TAB {
      index $\leftarrow$ code of the neighborhood configuration of p;
      If tab_prune[index] $= 0$ {
        $I(p) \leftarrow 2$; *fifo_add*(p);
      }
    }
  - Elimination of these branches:
    While *fifo_empty()* = *false* {
      $p \leftarrow$ *fifo_first*();
      index $\leftarrow$ code of the neighborhood configuration of p;
      If tab_prune[index] $= 0$ {
        $I(p) \leftarrow 0$; /* p is suppressed */
        For every pixel $p' \in N_G(p)$ {
          If $I(p') = 3$ {
            fifo_add(p'); $I(p') \leftarrow 2$;
          }
        }
      }
      else $I(p) \leftarrow 3$;
    }
  - Final scanning of TAB:
    For every pixel p in TAB {
      If $I(p) = 3$ then $I(p) \leftarrow 1$; /* production of a binary result... */

}

This algorithm is illustrated by Fig. 9. Like most of the algorithms based on queues of pixels [19, 20, 18], it is extremely efficient since, once the anchor points have been determined, it only requires:

1. one entire image scanning to initialize the queue,
2. one breadth-first scanning of the pixels of the set to skeletonize (the pixels with initial value 1),
3. a final pruning, where just a couple of pixels are considered.

It typically runs in less than 1 second on a *Sun Sparc Station1,* for a 256×256 image.



(a) original image *(b) distance function* *(c) level lines of this function*

(d) skeleton by maximal balls *(e) propagation of the "front"* *(f) final skeleton*

Figure 9 : Construction of the standard skeleton .

Now, since random access to the pixels is required, the algorithm is not adapted to specialized hardware. However, as explained in [19, chapter 7], even in this framework, the concept of anchor point is of great interest and a parallel version of the above algorithm can be proposed.

3 Various types of skeletons

We have already mentioned that in the above algorithm, any set can a priori be used as set of anchor points. We briefly mention below some of the most interesting skeletons that are derived from different anchor sets:

Minimal skeleton This skeleton was originally proposed by F. Meyer [12]. It is the smallest subset of the homotopic skeleton which:

- preserves the homotopy of the original set X,

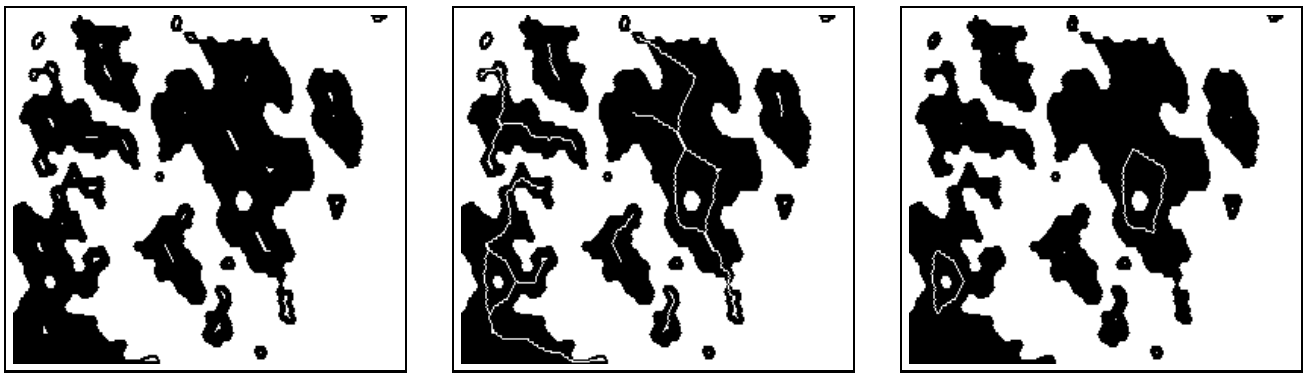
- contains the ultimate erosion $\mathrm{Ult}(X)$ of X.

Recall that the ultimate erosion of X is defined as the set of the *regional maxima* of the distance function of X, i.e., the set of the connected plateaus of dist_X such that every neighboring pixel has a *strictly* lower value [3, 21]. The minimal skeleton of X can be obtained here in a straightforward manner by taking $\mathrm{Ult}(X)$ as set of anchor points in our algorithm. This is illustrated on Fig. 10.a–b. Note that the ultimate erosion of a set can be efficiently determined by means of algorithms described in [19, chapter 5].

As may be observed on Fig. 10.b, the minimal skeleton exhibits far less branches than the standard one and is relatively insensitive to the noise on the boundaries of sets. Its extremities all mark some essential feature. Up to now, it has not been frequently used because of the lack of efficient algorithms to compute it, but it definitely provides in many cases more robust shape descriptors than the standard skeleton.

Homotopic marking It is also possible *not to impose any anchor points,* in which case the set X under study keeps being peeled as long as the homotopy is preserved. This results in the smallest possible homotopic subset of the standard skeleton, called *homotopic marking.* An illustration is given Fig. 10.c. One can observe that the homotopic marking of a simply connected component is reduced to a single pixel, approximately central. It thus provides an easy way to extract *centroids* which, unlike centers of gravity, have the advantage of being always located inside the original set. They are often referred to as *geodesic centers* and constitute one of the most immediate applications of homotopic markings.

These markings were classically obtained in morphology via parallel thinnings with structuring element D (see [7]) until stabilty is reached [3, 21]. The present method is much more efficient and also provides better results. Indeed, unlike D-skeletons, the determined homotopic markings are always of unit width [19, chapter 7].



(a) Ultimate erosion *(b) Minimal skeleton* *(c) Homotopic marking*

Figure 10 : More skeletons.

Geodesic skeletons The algorithm proposed in § 2 easily extends to the geodesic case [9]. Here, the set X to be skeletonized is supposed to be included in a mask M providing a new distance relation between pixels: the geodesic distance $d_M(p, q)$ between two pixels p and q of M is defined as the minimal length of the paths (if any) joining p and q and being included in M. The geodesic skeleton by maximal balls of X within M is given by the locus of the centers of the geodesic maximal balls of X. The present algorithm is easily adapted to this case.

Among other applications, geodesic skeletons can be used to connect particles inside a mask, as explained in [2]. Moreover, the frame (i.e., the entire image) is a frequently used geodesic mask. Indeed, using it amounts to assuming that the objects touching the edges of the frame are not completely included in it. In this case, some of the resulting skeleton branches will also touch these edges. This is ilustrated by Fig. 11. Naturally, minimal geodesic skeletons and geodesic homotopic markings are also easily obtained, although no one has used these transformations yet, since up to now, their computation was much too cumbersome!

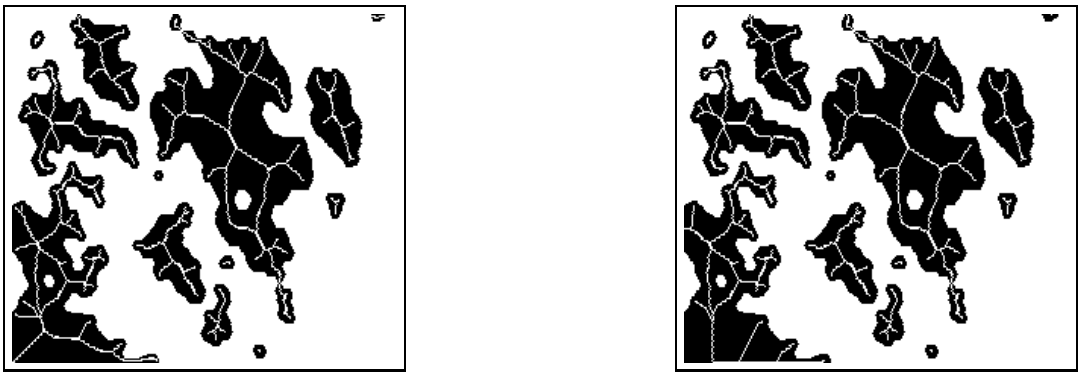


Figure 11 : *Standard skeleton (left) versus geodesic skeleton within the frame (right).*

Skeletons of increasing order One of the most immediate applications of the present algorithm is also the computation of smooth skeletons of various order: by taking as set of anchor points the centers of the maximal balls of radius greater than or equal to a given value r, one obtains a skeleton which is called *smoothed skeleton of order r* [19, chapter 7]. It is of course homotopic, and the extremities of its branches correspond to maximal balls of radius greater than or equal to r. The parameter r can be adjusted according to the level of noise on the boundary of X. In fact, the family of skeleton obtained in this manner provides a kind of hierarchical skeleton decomposition of X. Similarly, minimal skeletons of increasing order can be defined (see Fig. 12).

Notice that the skeleton of order r of X is different from what one would get by performing r pruning iterations of the standard skeleton. Here, the pruning is not done blindly, but is somewhat guided by the size of the balls along the skeleton, i.e., by the distance function. The result is in fact a homotopic superset of the skeleton of the opening of size r of X (see [17, 21]). This is illustrated by Figs. 12 and 13.

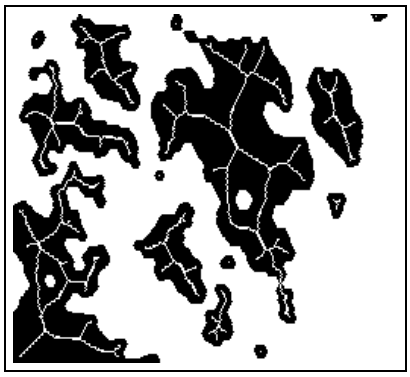
Exoskeletons, Skeletons by Influence Zones The skeleton of the background X^C can be computed using the same algorithm, yielding a transform called *exoskeleton* of X. Once again, minimal, geodesic and smoothed exoskeletons can be derived. It seems particularly interesting here to exclusively consider the geodesic case within the frame, which yields more consistent results. In this latter case, taking an empty set of anchor points yields the skeleton by influence zones (SKIZ) of X [8] (see Fig. 14). This is one of the most commonly used transformations of mathematical morphology.

4 Conclusion, Summary

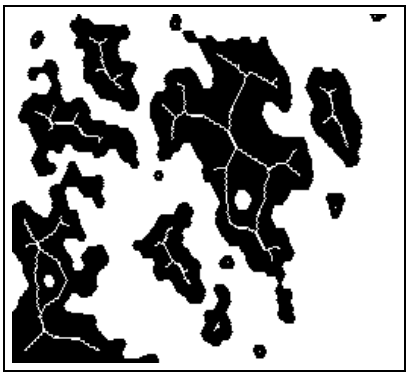
The various types of skeletons which have been proposed above only constitute a sample of what is made possible by the present algorithm. It provides a complete and consistent set of skeleton-like transformations, which can be finely adapted to each particular case. We therefore hope that thanks to this algorithm, the use of complex skeletonizations will become more common, and new fields of application will be opened.

We have shown that the present algorithm produces accurate results, since by definition, it contains the skeleton by maximal balls. We have also seen that no matter what set of anchor points is used, the computation times are extremely competitive. Together with its flexibility, this is probably the main advantage of the algorithm. In addition, like all algorithms based on queues of pixels, the present one can be easily adapted to any digital grid, and extended to images of arbitrary dimensions.

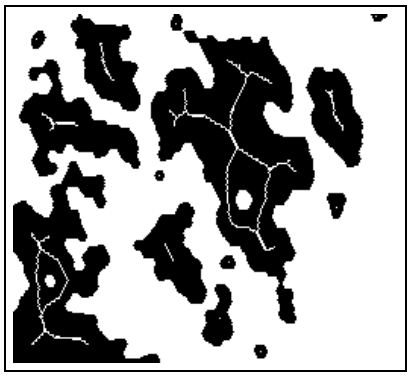
3-dimensional skeletons are currently being implemented with the present method. They should provide a useful set of tools for the analysis of confocal microscopic images. Lastly, the approach developed in this paper has been combined with the watershed algorithm described in [22], yielding a family of efficient grayscale skeleton algorithms. It will constitute the topic of further publications.



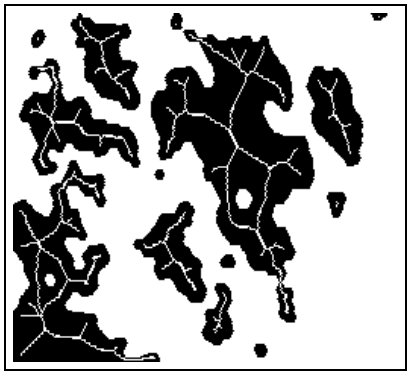
(a) skeleton of order 4



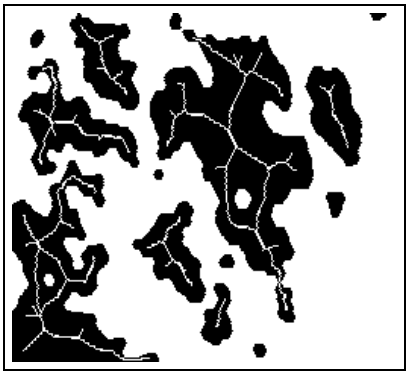
(b) skeleton of order 8



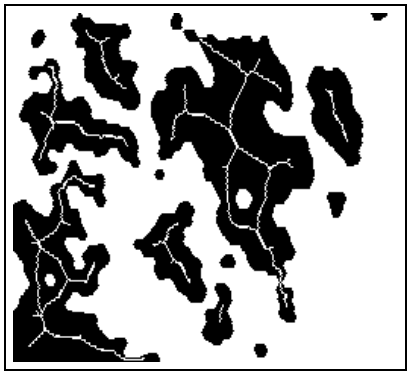
(c) skeleton of order 12



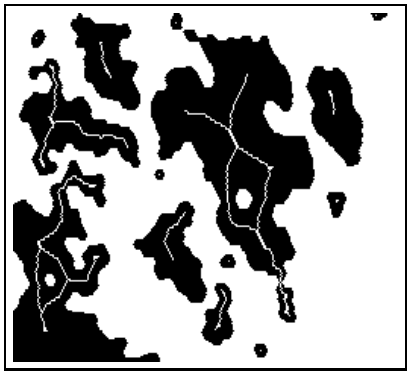
(d) skeleton after 4 pruning steps



(e) 8 pruning steps



(f) 12 pruning steps

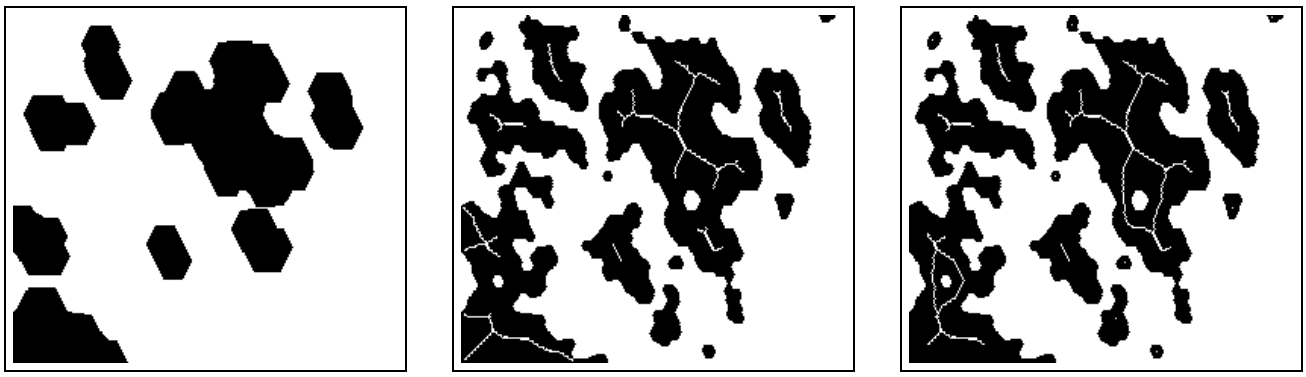


(g) minimal skeleton of order 5

(h) minimal skeleton of order 10

(i) minimal skeleton of order 15

Figure 12 : Various kinds of skeletons...



(a) hexagonal opening of size 12 *(b) skeleton of the opening* *(c) skeleton of order 12*

Figure 13 : *Skeleton of the opening versus skeleton of a given order (note that the opening and its skeleton are here* geodesic within the frame*).*

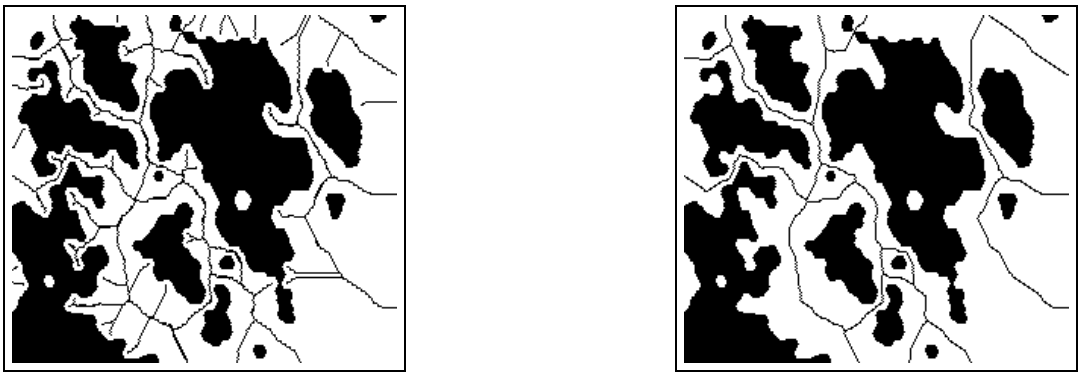


Figure 14 : *Exoskeleton (left) and skeleton by influence zones (right).*

5 Acknowledgments

This work was supported in part by *Dassault Electronique* and the *National Science Foundation* under Grant MIPS–86–58150, with matching funds from *DEC* and *Xerox*. The author is very grateful to Gaile Gordon for useful comments and suggestions.

6 References

1. C. Arcelli, L. Cordella, and S. Levialdi. From local maxima to connected skeletons. *IEEE Trans. Pattern Anal. Machine Intell.*, 2:134–143, 1981.
2. S. Beucher. *Segmentation d'Images et Morphologie Mathématique*. PhD thesis, Ecole des Mines, Paris, June 1990.
3. S. Beucher and L. Vincent. Introduction aux outils morphologiques de segmentation. In *Traitement d'Images en Microscopie à Balayage et en Microanalyse par Sonde Electronique*, pages F41–F43. ANRT, Paris, Mar. 1990.
4. H. Blum. An associative machine for dealing with the visual field and some of its biological implications. In E. Bernard and M. Kare, editors, *Biological Prototypes and Synthetic Systems*, pages 244–260. Plenum Press, New York, 1962. Proc. second Annual Bionics Symposium, Cornell University, 1961.
5. G. Borgefors. Distance transformations in digital images. *Comp. Vis., Graphics and Image Processing*, 34:334–371, 1986.
6. L. Calabi and W. Harnett. Shape recognition, prairie fires, convex deficiencies and skeletons. Technical Report 1, Parke Math. Lab. Inc., One River Road, Carlisle MA, 1966.
7. M. Golay. Hexagonal pattern transforms. *IEEE Trans. on Computers*, 18(8), 1969.
8. C. Lantuéjoul. Issues of digital image processing. In R. M. Haralick and J.-C. Simon, editors, *Skeletonization in Quantitative Metallography*. Sijthoff and Noordhoff, Groningen, The Netherlands, 1980.
9. C. Lantuéjoul and F. Maisonneuve. Geodesic methods in quantitative image analysis. *Pattern Recognition*, 17(2):177–187, 1984.
10. G. Matheron. Examples of topological properties of skeletons. In J. Serra, editor, *Image Analysis and Mathematical Morphology, Volume 2: Theoretical Advances*. Academic Press, London, 1988.
11. G. Matheron. On the negligibility of the skeleton and the absolute continuity of erosions. In J. Serra, editor, *Image Analysis and Mathematical Morphology, Volume 2: Theoretical Advances*. Academic Press, London, 1988.
12. F. Meyer. Skeletons and perceptual graphs. *Signal Processing*, 16(4):335–363, 1989.
13. A. Montanvert. *Contribution au Traitement de Formes Discrètes. Squelettes et Codage par Graphe de la Ligne Médiane*. PhD thesis, Université Scientifique, Technologique et Médicale de Grenoble, France, 1987.
14. A. Rosenfeld and J. Pfaltz. Sequential operations in digital picture processing. *J. Assoc. Comp. Mach.*, 13(4):471–494, 1966.
15. A. Rosenfeld and J. Pfaltz. Distance functions on digital pictures. *Pattern Recognition*, 1:33–61, 1968.
16. M. Schmitt. *Des Algorithmes Morphologiques à l'Intelligence Artificielle*. PhD thesis, Ecole des Mines, Paris, Feb. 1989.
17. J. Serra. *Image Analysis and Mathematical Morphology*. Academic Press, London, 1982.
18. L. van Vliet and B. J. Verwer. A contour processing method for fast binary neighbourhood operations. *Pattern Recognition Letters*, 7:27–36, Jan. 1988.
19. L. Vincent. *Algorithmes Morphologiques à Base de Files d'Attente et de Lacets: Extension aux Graphes*. PhD thesis, Ecole des Mines, Paris, May 1990.
20. L. Vincent. New trends in morphological algorithms. In *SPIE/SPSE Vol. 1451, Nonlinear Image Processing II*, pages 158–169, San Jose, CA, Feb. 1991.

21. L. Vincent and S. Beucher. The morphological approach to segmentation: an introduction. Technical report, Ecole des Mines, CMM, Paris, 1989.

22. L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Trans. Pattern Anal. Machine Intell.*, 13(6):583–598, June 1991.

23. Y. Xia. Skeletonization via the realization of the fire front's propagation and extinction in digital binary shapes. *IEEE Trans. Pattern Anal. Machine Intell.*, 11(10):1076–1086, 1989.