SHADOW OF A MULTI DIMINUTION BINARY IMAGE RESOLUTION WITHOUT EXPLICIT IMAGE SMOOTHING

Narasimha Rao Jonnadula[1*], Dr.Harinatha Reddy B[2*], Dr.Raja Prakash Rao P[3*]

1. *II.M.Tech-CSE, TRREC , narasimharao.jonnadula@gmail.com.*
2. *Professor, Dept of CSE, TRREC,*
3. *Professor & Head, Dept of CSE, TRREC*

Keywords:
Cloud computing, information security, Internet hacking, Server as a Server, long term evolution.

Abstract: *Three methods described in this paper deal with the problem of multi-resolution problem of skeletonization without explicit image smoothing. The methods are: (1) the removal of "tips" during the construction of a skeleton or after a skeleton has been obtained, (2) the removal of skeleton branches during the first few iterations or once only at a predefined iteration, (3) removal of all skeleton points from the nearest node to the remaining image at a predetermined iteration. Comparison with some smoothing methods is also presented.*

1. INTRODUCTION

A skeleton of a two dimensional binary image (a finite subset of the discrete plane 2') is an image region that is in some sense (i) thin, (ii) "central" to the binary image, (iii) preserves its homotopy, (iv) preserves some other aspects of its shape, e.g. reconstruct ability [lo]. Most skeleton algorithms are sensitive to noise in the image boundary. A noisy image results in a skeleton with many spurious branches, which make further analysis more difficult. Skeletons obtained at different resolutions may be needed to detect interesting regions [4]. The higher the resolution (smaller the scale) is, the more detail of the curvature changes in the image boundary or the smaller convexities of the original image regions can be observed in the skeleton. Therefore multi-resolution ability is an important practical requirement for a skeletonising procedure.

Noise can be suppressed by cleaning the image [2] or by removal of contour pixels which are considered as noise from analysis of the contour curvature [1]. Multi-resolution can be achieved either as a part of skeletonising procedures or by separate pre-processing.

***Narasimha Rao Jonnadula**
II.M.Tech-CSE, TRREC

Approaches that have been used include using a symmetrical low pass filter with variable bandwidth[4], or by morphology smoothing operations, i.e. opening or closing using a disc structuring element with an appropriate radius [14], or by an approximation of the original image considering both boundary and region centered descriptions at different scales[3]. Other methods rely on carefully selecting the skeleton points: the skeleton starts only from points whose convexity is greater than a given threshold [4, 12], from local maxima and "balanced saddle points'' [9], or according to the "propagation velocity" [5]. Polygonal approximation of the original contour is also used [8].

In this paper we describe the integration of efficient multi resolution procedures with a general skeletonisation program [10]. It allows (1) choice of how the noise in the skeleton is suppressed, and (2) control of the required resolution

2 BRIEF DESCRIPTION OF SKELETON ALGORITHM

We described an algorithm [10] which explicitly separated the two major aspects of skeletonisation; (i) the identification of points critical to shape representation, (ii) the identification of further points necessary to preserve homotopy. Decoupling the shape and topology-preserving aspects of skeletonisation permits the implementation of a variety of distance functions, unlike almost all other skeleton methods in the literature. One benefit of this approach to the skeletonization problem is that there is no need to modify the original image in order to achieve multi-resolution.

In the skeleton procedure sets of points critical to shape representation are found by eroding the original image *I* with a nested sequence of structuring elements E_j, where E_O = { 0 } (origin of 2') and E_i c E_i+*l* , corresponding to a generalized distance transform. At each iteration *i,* the set of shape-related points M_i is defined by $I_i = I\theta E_i, M_i = I_i(I_i \oplus D)$, and D is a suitable small structuring element. Depending on *D,* the set of points $\cup_{i-1}^{m} M_i$ corresponds more or less closely to the local maxima of the distance from the background of the original image [lo]. Points needed in addition to those of M_i , in order to preserve homotopy, may be found at each iteration by a search restricted to the "shell" $S_i = I_i \setminus I_{i+1}$. By choosing appropriate E_i and *D,* the algorithm is capable of producing a variety of skeletons corresponding to different distance functions, including city block, chess board, octagonal, Euclidean and even asymmetric ones. If for all $i, E_i \oplus D \subseteq E_{i+1}$, then the original image can be reconstructed from the skeleton

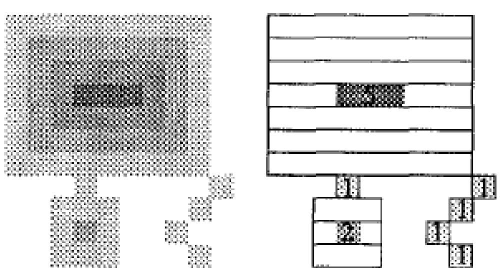

Figure 1: Shells, and labeled M points.

[10]. Fig 1 illustrates the definition of shell (shaded) and *M* points (labeled with the shell number).

It has been shown [13, 101 that if each point removed from an image satisfies a simple condition (the "Hilditch condition" *[6])* with respect to the remainder of the image at that time, then topology is preserved. Point *p* satisfies the Hilditch condition with respect to a current image region U if and only if the set of 8-neighbours of *p* which belong to *U* is non-empty and simply connected. If only points which satisfy the Hilditch condition are removed at each iteration, the resulting skeleton will have the same homotopy as the original.

Since points in *Mi* have the same distance *I* (corresponding to the metric used and the erosion sequence) to the background and represent the significant shape of *I* related to *Ei* (e.g. *Ei* can be placed at *Mi* and contained in the original image), the omission of points *Mi* from the skeleton means that the resolution will reduce to the extent that the influence of fine variation in shape whose scale is smaller than *E,,,* will not appear in the skeleton. In other words, when the image is reconstructed from the skeleton that scale of detail will not reappear. However we cannot achieve this simply by removing all U:, *M;* for a particular scale *m,* since this may change the connectivity of the original image. We can proceed in two ways. One is based on the removal of tips. The skeleton preserves the topology of the original image, and the removal of tips will not change the connectivity. The other possible method is based on a check of connected components of *Mi* in each interaction, or detractively of *Mi* at iteration *m.* Some (or all) components will be removed so long as the removal does not change the connectivity of the original image.

3 MULTI-RESOLUTION CHOICE

For simplicity we assume that the skeleton is already one pixel thin' and that the resolution needed is *m* (an integer, predefined by user)

The following definitions are used in this context.

- Node: a node in a skeleton is where branches occur i.e. a point with more than two neighbors which are not themselves 4-connected.
- Tip: a tip of a skeleton is a point with only one neighbor in the skeleton.
- Depth we use two different definitions of the depth of a skeleton point: (1) the shell number *i* to which the point belongs, i.e. the distance to the background, or (2) *i* plus the distance between the point in question and the nearest tip.
- Elongated region: a compact set of many points in a image region with similar, and short distances to the background.

The two definitions of "depth" will result in different skeletons. The first depends only on the distance to the background. The second will favor elongated regions. When the resolution decreases *(m* increases) the skeleton points representing an elongated part will disappear gradually instead of disappearing all at once when *2m* reaches the thickness of the region

We have investigated three ways to obtain a skeleton with resolution *m.*

METHOD 1. In the first method tips are removed. It can be applied during each of the first *m* iterations or after the skeleton has been obtained. Since a tip is always simply connected to its single neighbor, it satisfies the Hilditch condition, and thus the topology will be maintained after it is removed.

In the former case, the second depth definition is used z. All tips are removed in the first *m* iterations, if the eroded image component has not vanished. The result of this approach is similar to that of omitting the Hilditch's requirement (3) [6], during the first *m* iterations.

In the latter case, the removal procedure can be iterative i.e. all tips will be removed at each of *m* iterations; or recursive i.e. a tip is removed and then its neighbor until *m* points are removed or a node is reached, and all tips in the skeleton have been treated in the same fashion. When alternatively the distance to the background of the tips is compared with *m* (depth definition 2), the result may be slightly different.

This method is easy to implement but the latter needs extra computation time because part of the skeleton will be processed twice: once for inclusion, and again for exclusion. Fig 2 shows an example.

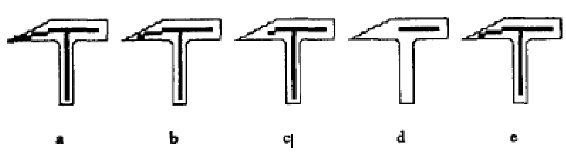

Figure 2: Skeleton of a hammer shape obtained by method 1, with different resolutions m, (a) m = 0, (b) m = 1, (c) m = 2, (d) m = 3, depth is defined to be the distance to the background, (e) m = 3, depth is defined to be the distance to the background plus the distance to the nearest tip

METHOD 2. The Second method uses the first depth definition, and again can be used iteratively or once only. Let K_{i-1} be the, resulting skeleton points from previous iterations, and X_i^j be the j-th connected component of $M_i \cup K_i$. At each of the first m iterations, X_i^j is deleted (and does not therefore form part of K_i) so long as this does not alter the topology of $M_i \cup K_{i-1} \cup J_i \cup I_{i+1}$ (J_i is a set of points in shell S_i necessary to maintain the connectivity at iteration *i* [lo]). The tips in the remaining X_i^j are also deleted recursively until node@) are reached. Fig 4 shows skeletons obtained by this method.

The skeleton obtained by this method can match the result from a smoothed image obtained by applying an opening to the original, except that it always maintains the original connectivity (see fig. 5 b and *f),* This method is very powerful, especially when the image is noisy. Since lots of potential skeleton points are removed from subsequent iterations, it may actually make the skeleton procedure faster. The results may be unsatisfactory when there is no prior knowledge about the initial image and the resolution chosen is too low. For example, the skeleton of an image, whose boundary is formed by arcs of two circles with different radii and common tangent lines, will shrink towards the centre of the larger arc while the scale is greater than the smaller radius (when m > *radius,* see fig. 3 b).

METHOD 3. The third method is applied once, only at the end of iteration *m.* A search starts from skeleton points i n, S, which directly connect with *Ina+l,* to wards

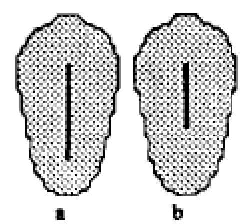

Figure 3: An example of skeleton obtained by method 2 with scale m = 0 and m = 11. If the skeleton point removal depends soly on the distance to the background, it will shrink towards the thicker part.

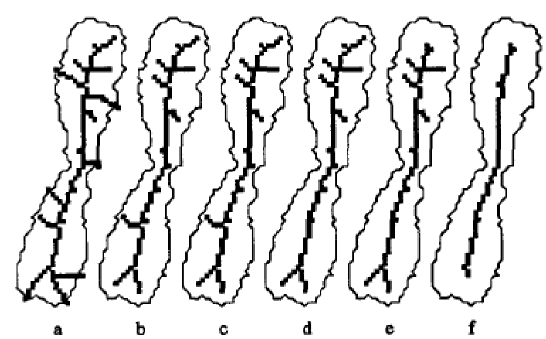

Figure 4: Euclidean skeletons of a chromosome with different resolutions using method 2. (a) m = 0, (b) m = 1, (c) m = 2, (d) m = 3, (e) m = 4, (f) m = 5.

Their neighbors until either a tip or a node in the skeleton is reached. A node is marked. If there is no node, the whole skeleton branch just being searched remains; otherwise every branch which starts from a tip and ends at a node is deleted until a marked node is reached. This method has a bias, preferring to remove skeleton branches at noisy regions, and to maintain the skeleton unchanged at a smoother region. For instance, the skeleton of the previous example will remain stable even if m is greater than the smaller radius.

4 TIMING AND BRIEF COMPARISON

The following table shows the timing for the three methods described above together with two smoothing methods. The first smoothing method is opening by a large disc (or square for chess board metric); the scale corresponds to the radius used in the operation. The second is boundary smoothing, in which 2 x m + 1 points smoothing is used. The skeleton procedure is written in the C programming language and was tested on a SUN SPARCstation 1. A test image size 127 x 147 pixels was used. The figures in the table for smoothing methods include time both for smoothing and skeletonising. The image and its Euclidean skeletons in different resolutions from different methods are shown in fig. 5

Test image of fig. 5, timing in ms				
metric	chess board		Euclidean	
scale "m"	5	10	5	10
opening	200	220	381	340
boundary smoothing	360	430	456	439
method 1(1)	164	167	440	419
method 1(2)	147	163	426	406
method 2 (1)	141	158	395	420
method 2 (2)	142	148	384	391
method 3	166	176	439	413
scale 0	142		408	

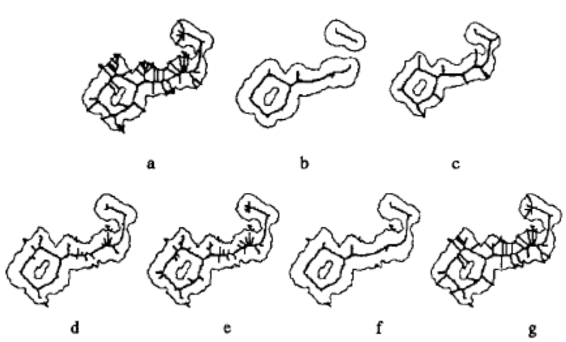

Figure 5: Euclidean skeletons of the test image at resolution 10: (a) original, (b) opening, (c) boundary smoothing, (d) method l(l), (e) method 1(2), (0 method 2, (g) method 3.

5 SUMMARY AND CONCLUSIONS

In the above test, the boundary smoothing method is the slowest. The reason is that before applying a boundary smoothing method, the image has to be transformed from an interval object [11] to a polygon representation, and then changed back after being smoothed.

The opening procedure is among the fastest in the above test but it may break the connectivity of the original image (fig. 5b).

The first method described above is easy to implement and suitable for elongated images, but less efficient when long whisker-like noise is present on the image boundary.

The second method can achieve the same result as the opening procedure, has the advantage of maintaining the topology of the original image, and is usually fast.

The third method is only useful in special cases.
An interesting example is the reconstruction of the image from its low resolution skeleton which can smooth the image [7]. In our method the connectivity is preserved (see fig *6)*

ACKNOWLEDGEMENTS

I would like thank my colleague Jim Piper and Denis Rutovitz for their advice.

REFERENCES

1) C Arcelli. Pattern Thinning by Contour Tracing. Comp. Graph. and Image Proc. 17,130-144, 1981.

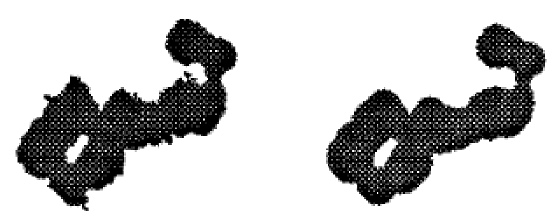

Figure 6: The reconstruction of the original image (left) from its Euclidean skeleton (right) at resolution 10. The image is smoothed and its connectivity is preserved.

2) C Arcelli, G Sanniti de Baja. A width-independent fast thinning algorithm. IEEE PAMI-7,464-474, 1985.
3) M. Brady and H. Asada. Smoothed Local Symmetries and Their Implementation. The Intemationd Journal Robotics Research. Vol3,36-61, Fall 1984.
4) A.R. Dill, M. D. Levine and P. B. Noble. Multiple Resolution Skeletons. IEEE PAM1-9,495-504, 1987.
5) R. 0. Duda and P. E. Hart. Pattern Classification and Scene Analysis. New York: Wiley, 1973.
6) J Hilditch. Linear skeletons from square cupboards. In: B Meltzer, D Michie (eds) Machine Intelligence 4, Edinburgh University Press, 404-420, 1969.
7) S . B. Ho and C. R. Dyer. Shape Smoothing using medial axis properties. IEEE PAM1-8,512-520, 1986.
8) U Montanari. Continuous skeletons from digital images. J. ACM 16,534-549, 1969.
9) C W Niblack, D W Capson and P B Gibbons. Generating skeletons and Centerlines from the Medial Transform. IEEE ICPR-B, 881-885, 1990.
10) L. Ji, J. Piper. Fast homotopy-preserving skeletons using mathematical morphology. IEEE PAMI. in press, 1992.
11) J Piper, D Rutovitz. Data structures for image processing in a C language and Unix environment. Patt. Recog. Letts. 3, 119-129, 1985.
12) S Riazanoff, B Cervelle, J Chorowicz. Parametrisable skeletonization of binary and multilevel images. Patt. Recog. Letts. 11,25-33, 1990.
13) A Rosenfeld, A C Kak. Digital Picture Processing. Academic Press, New York, 1976.
14) J Serra. Image analysis and mathematical morphology. Academic Press, London, 1982.