Pattern Recognition Letters 13 (1992) 237-243
North-Holland

April 1992

Ridge points in Euclidean distance maps

Carlo Arcelli and Gabriella Sanniti di Baja

Istituto di Cibernetica, CNR, Via Toiano 6, 80072 Arco Felice, Naples, Italy

Received 13 February 1991

Abstract

Arcelli, C. and G. Sanniti di Baja, Ridge points in Euclidean distance maps, Pattern Recognition Letters 13 (1992) 237-243.

Two types of ridge points are identified on the Euclidean distance map of a digital object. From this set, which is connected, the skeleton of the object is derived as a unit width subset. The use of the Euclidean distance map guarantees obtaining a skeleton with structure sufficiently stable under object rotation, and placed where it is expected to be found according to human intuition.

Keywords. Euclidean distance map, ridge point, skeleton.

1. Introduction

A distance map is the result of a labeling process which assigns to the pixels of a given object values equal to the distance of the pixels from the complement of the object. The distance can be computed according to different functions.

Distance maps can be interpreted as landscapes of islands where the label of every pixel indicates the height of the region which is the continuous counterpart of the pixel itself. A peculiarity of a distance map with respect to a true natural landscape, is that some constraints exist on the label of neighboring pixels. In fact, the distance map can be understood as generated by a wavefront, propagating with constant speed from the complement of the object towards the interior of the object. Pixels reached at the same instant of time have the same label, and correspond in the landscape to pixels with the same height. Since labels of neighboring pixels can change only within a limited exent, the landscape is characterized by gentle slopes converging towards at most two pixels wide ridges and peaks. Moreover, plateaux cannot exist and craters exist only if they include one lake.

Intuitively, detecting ridges and peaks is a straightforward way to find the skeleton of the object, understood as generated by the grassfire transformation [3]. Several algorithms have been proposed in the past to skeletonize the distance maps computed according to the chessboard, the city-block and weighted distances, e.g., [1,3,8]. These algorithms identify on the distance map skeletal pixels which, at least implicitly, can be understood as placed in correspondence of ridges and peaks. The main drawback of the skeletonizing algorithms using the chessboard or the city-block distance is that the transformation is remarkably not invariant under object rotation. In fact, cardinality, spatial distribution and label of the skeletal pixels may change drastically with object orientation. Better results can be obtained by using weighted distances, as these provide closer approximations to the Euclidean distance. The approximation degree depends on the adopted weights and on the size of the neighborhood from which distance information is derived [5].

Euclidean distance maps (EDM, for short) should be the most suited when invariance under rotation is desired. However, their use, when working with conventional sequential computers, has been up to now hampered by the involved computational burden. In fact, while the chessboard, the city-block and the weighted distance maps can be computed within two sequential raster scans [5], four inspections were required to get EDM [7]. Recently, an algorithm has been proposed which requires only three inspections [11,12]. Thus, the difference in cost to obtain EDM and the weighted distance map has been reshuffled, and it could be worthwhile extracting the skeleton on EDM.

We extend to the Euclidean case, the skeletonization method based on ridge and peak detection. Two types of pixels are extracted from EDM. Pixels of the first type (peaks and strong ridge points) are extracted parallelwise by using local operators. Pixels of the second type (weak ridge points) are extracted sequentially by growing paths in the direction of the steepest gradient.

The proposed algorithm has a reduced computational cost. In fact, peaks and strong ridges are found within one raster scan of EDM. Within the same scan, weak ridge points are sequentially extracted as elements of paths, starting from strong ridge points and growing towards either peaks or strong ridge points. The resulting set is connected and centered within the object, as the skeleton is required to be, and can be interpreted as a stick like version of the object. Indeed, due to the discrete nature of the digital plane, the set is likely to be two pixels wide (even more, in few cases), so that reduction to unit width has to be achieved after peaks and ridges have been detected. To this purpose, a second raster scan is employed. The obtained set has its pixels labeled with the Euclidean distance from the complement of the object. Thus, width information is retained, and the skeleton can be used for analyzing the shape of objects characterized by nonconstant width.

2. Peaks and ridges

Let us consider a two-dimensional object F digitized on a square array, and denote by B the complement of F. For the sake of simplicity, we assume that F consists of one 8-connected component, regardless of its connectivity order, and does not touch the frame of the array. Moreover, we assume that F has undergone a cleaning process devoted to fill noisy holes. For instance, this can be achieved by assigning to F any pixel of B surrounded by at least six neighbors belonging to F. This cleaning cannot be demanded to the pruning process, which is generally applied to any skeleton before its use in real tasks. In fact, once the not significant holes of F have been mapped into the loops of the skeleton, the structure of the skeleton turns out to be irreversibly damaged.

An early attempt to skeletonize EDM was mentioned in [7], but that skeleton, not generally connected, is not suited to deal with description tasks. The algorithm described in [9] allowed one to obtain a more useful skeleton, though, as pointed out in [10], computational effectiveness could be achieved only by using special hardware. Recently, it seemed that a simple and effective way to compute the Euclidean skeleton could be available [13]. Indeed, notwithstanding the authors' claim, the proposed operations, defined in the 3×3 neighborhood and parallelwise applied to the pixels of EDM, do not guarantee maintainance of connectedness. Moreover, erroneous skeleton loops, which surround one pixel, are created not in correspondence with holes of the object. A minor drawback, pointed out also by the authors, is the existence of a number of skeleton branches significantly larger than the expected one. The performance of the algorithm is shown in Figure 1. The

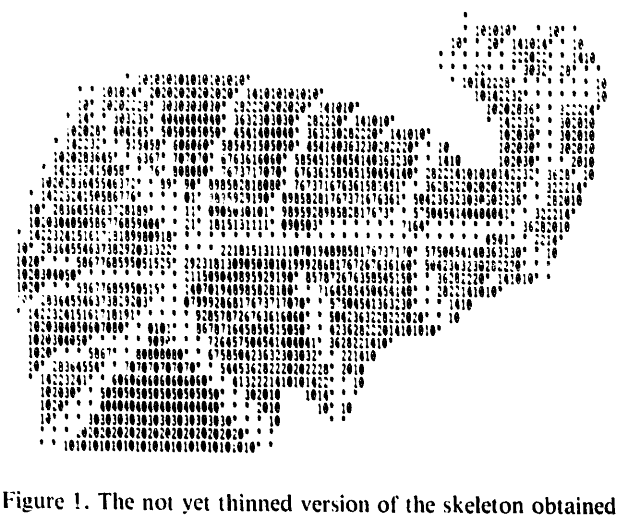

Figure 1. The not yet thinned version of the skeleton obtained by algorithm [13] . The Euclidean distances are represented by rounding their actual value times ten.

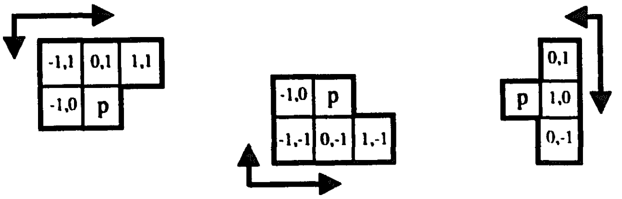

Figure 2. Masks used in [12] for computing EDM within three raster scans.

printout refers to the skeleton as obtained before the application of the last step of the algorithm, tailored to obtain unit skeleton width.

In this paper, EDM is computed by the three-raster-scan algorithm described in [12]. The errors which may affect the resulting map are due to the sequential process, as pointed out in [14], and can generally regarded as negligible. The three raster scans are respectively performed row-by-row from top-left to bottom-right on the set *F*, row-by-row from bottom-left to top-right, and column-by-column from top-right to bottom-left on the successively resulting sets. The masks employed within each scan are depicted in Figure 2, as well as the relative scanning mode. We bypass the square root computation and let the pixels be labeled with the square value of the Euclidean distance, since the operators we employ are equally effective. As a result, the skeletonizing algorithm uses mainly integers. In what follows, unless differently stated, small letters will be used to indicate both pixels and their associated labels.

Ridge points and peak points are found on EDM, sequentially inspected in forward raster fashion. These pixels are marked on EDM, by multiplying their label by -1. In this way, distance information is preserved.

We regard as strong ridge points the pixels belonging to ridges oriented along any of the horizontal, vertical and diagonal directions. Accordingly, a strong ridge point is a pixel *p* of EDM having two neighbors with smaller label, diametrically opposed along any of the previous directions. The employed 1×3 operators are shown in Figure 3(a)–(d). Since on the digital plane two pixels wide components of ridge points are likely to exist, also the 1×4 operators shown in Figure 3(e)–(h) are used. In this case, also the equilabeled neighbor of *p* is marked. The eight operators are all applied to each pixel in EDM, since a pixel can be embedded in more than one of the shown local configurations.

If *p* is detected as a ridge point by one of the 1×4 diagonal operators, marking it and its equilabeled neighbor could produce unwanted thickening when either one of, or both the neighbors dashed in Figure 3(g)–(h) are ridge points. In this case, the two dashed pixels are checked to verify whether any of them can be marked in place of *p* and of its equilabeled neighbor in the 1×4 mask. This check only requires label comparison: each of the dashed neighbors of *p* is marked provided that it has label larger than *p*. When either one or both the neighbors of *p* are marked, the coordinates of *p* and of its equilabeled neighbor are stored, since not marking these pixels may cause creation of erroneous skeleton loops. At the end of the raster scan, the pixels whose coordinates have been recorded are directly accessed and marked if all their 4-adjacent neighbors result to be already marked. In this way skeleton loops, surrounding just one pixel not in *B*, are filled in, and topology is preserved.

Note that for the pixels labeled '1' on EDM, the operators of Figure 3 are not sufficient to guarantee that the resulting skeleton is connected. The operators to be used, illustrated in Figure 4, are the ones introduced in [2] to identify on an 8-connected curve all the pixels located where the curve folds upon itself or intersects another 8-connected curve. These pixels can be easily understood as ridge points. Since the pixels labeled '1' constitute an

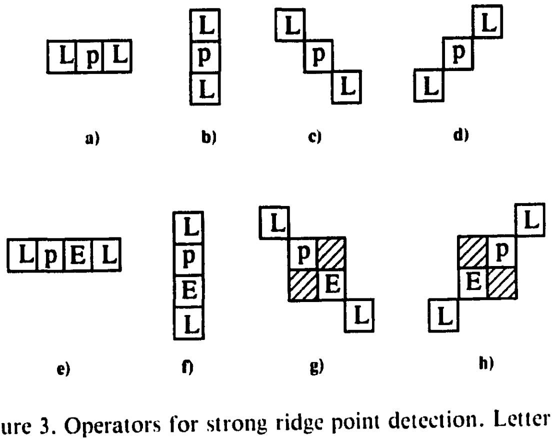

Figure 3. Operators for strong ridge point detection. Letters E and L denote labels respectively equal to and less than the label of *p*.

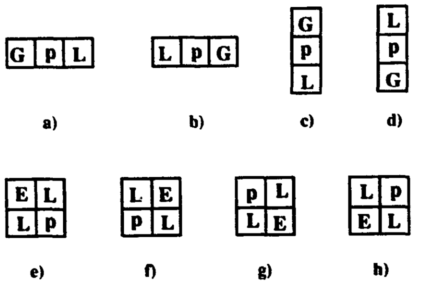

Figure 4. Ridge point detection operators for pixels labeled '1' in EDM. Letters G, L and E respectively denote pixels labeled more than, less than and equal to p.

8-connected curve in EDM, the previous operators successfully apply. Each of these pixels is marked as ridge point if none of the operators of Figure 4 (a)-(d), or at least one of the operators of Figure 4(e)-(h) matches the local configuration of EDM, where p is embedded.

Peaks can be seen as pixels surrounded by neighbors with smaller height. Thus, a pixel should be detected as a peak point if in its 3×3 neighborhood all the pixels have smaller label. However, sets including up to four equilabeled pixels are likely to exist, which can be seen as constituting the same peak. Then, we must accept as peak point any pixel having no neighbor with greater label. Since to verify the peak point condition the same operators devised for detecting strong ridge points are sufficient, in what follows the term strong ridge point will be used to identify also any peak point.

The set of the strong ridge points is generally not connected. This is due to the reduced number of directions along which ridge points are searched and to the small operator size. On the other hand, using a more sophisticated ridge detector could create unwanted thickening of the set of the ridge points without necessarily ensuring the connectedness of the set. Indeed, an effective criterion to achieve connectedness is to grow paths along the directions of the steepest gradient in EDM. Accordingly, paths are started from strong ridge points and terminate on strong ridge points. Pixels accepted as elements of paths are termed weak ridge points.

The first pixel in a path is the neighbor q of a strong ridge point p, such that the gradient of q to p is maximum. The gradient is computed as follows:

$$\text{grad}(q) = w_i(q-p)$$

where $w_i = 1$ or $w_i = 1/\sqrt{2}$, depending on whether q is horizontally/vertically or diagonally adjacent to p, and p and q should be understood as the square root of the corresponding labels.

The next pixel in the path is similarly found, if any, by looking at a subset of the neighbors of q, coherently with the direction $p \rightarrow q$.

Paths are grown towards all the possible directions, within the raster scan during which strong ridge points are detected, and continue as far as neighbors with increasing label are found. Really, paths growing towards regions already inspected by the raster scan terminate as soon as the neighbor with maximal gradient results to be already marked.

Since paths are grown also towards positions not yet visited by the raster scan, it may happen that pixel p is already marked, when inspected. Before continuing the raster scan, the operators of Figure 3(e)-(h) are applied to p, to assign to the ridge also its possibly existing equilabeled neighbor. In fact, disregarding such a neighbor may create an erroneous loop.

To gain unit width, one raster scan is generally sufficient. However, the shape of the resulting skeleton is not the optimal one, since superfluous pixels can still exist, due to the presence of 4-internal pixels in the set of the ridge points R. Moreover, if R includes configurations consisting of eight paths converging towards the same position, the skeleton will include 8-internal pixels. Note that, in presence of these irreducible sets, any skeletonizing algorithm equally fails.

The operators employed to gain unit width are shown in Figure 5, where letters r and b respectively denote pixels of R and of its complement, and at least one dashed pixel is in the complement. Marking is removed from p if at least one of the operators successfully applies. The sequential

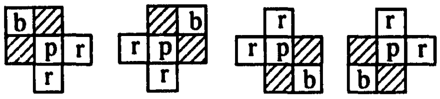

Figure 5. Operators for unit width reduction.

application of the operators to the ridge points guarantees maintainance of topology and prevents shortening of branches. Creation of holes is prevented by requiring that at least one dashed pixel does not belong to R, and connectedness is maintained, since b is also required to be in the complement of R. Shortening is prevented because the operators fail when applied to pixels having just one neighbor in R, the end points.

During the raster scan, track is kept of pixels which can be understood as located on the tip of skeleton branches, i.e., the end points. Candidate end points are the ridge points which survive the unit width reduction process and, when examined, have just one 8-connected component of ridge points in their 3×3 neighborhood. The set of the recorded pixels does not strictly coincide with the end points of the skeleton. In fact, the number of 8-connected components of ridge points in the neighborhood of a pixel p is likely to increase after some neighbor of p, following p in the raster inspection, is removed from R. Storing the coordinates of the candidate end points will allow one to drive a successive pruning step, with the aim of deleting not significant branches mostly due to both contour noise, and to the simple thinning procedure described above. Branches originating from end points should be traced and possibly pruned. A clever pruning should take into account the structural relevance of a skeleton branch, and evaluate the reconstructability power of the branch itself. In this respect, identification of the local maxima is crucial. At the moment, we limit pruning to unitary length branches, certainly not significant. The easiest way to implement this pruning, is removing among the previously recorded pixels, all those still having just one 8-connected component of ridge points in their 3×3 neighborhood, even if this will slightly shorten also the branches remaining in the skeleton.

In Figure 6(a) the set R is shown as a set of stars over EDM, while the obtained skeleton is shown in Figure 6(b).

3. Discussion and conclusion

The computational cost of the algorithm is rather modest. The skeleton is found after two inspections of EDM: the first devoted to the detection of the ridge points, the second used to gain unit width. A path growing process is used within the first scan, but its cost can be disregarded with respect to the overall cost of the process. Also negligible is the cost of filling erroneous loops and of pruning, which are respectively done at the end of the first and of the second inspection of EDM, by directly accessing suitable pixels stored to this purpose. A counterpart of the computational convenience is the somehow low pictorial quality of the resulting skeleton. This is particularly due to the one-pass sequential process employed to gain unit width. In fact, the application of the operators of Figure 5 is likely to produce some jaggedness and/or to prevent removal of some superfluous pixels. Our choice of accepting a rough skeleton is

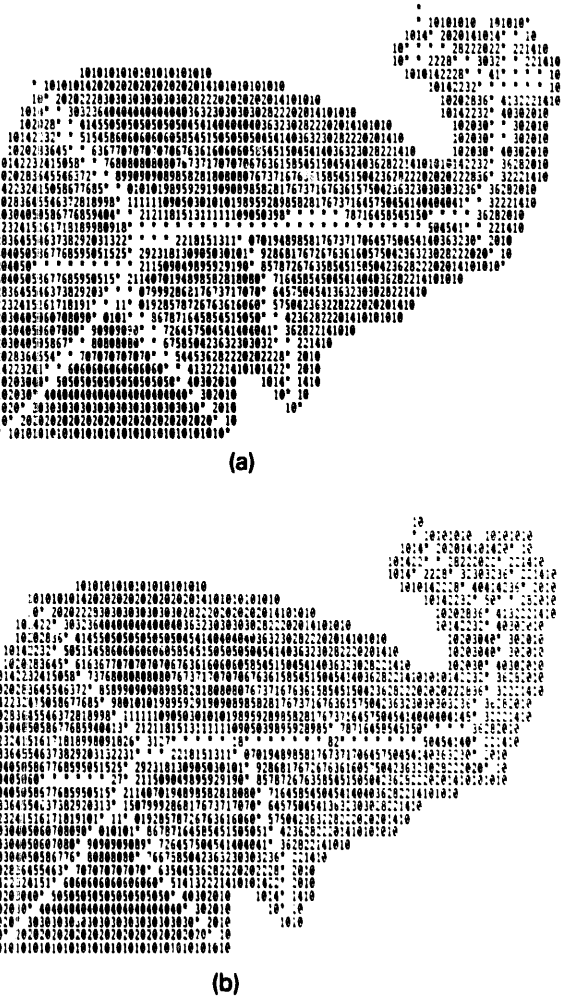

Figure 6. Set of the ridge points on EDM (a), and the resulting skeleton (b).

mainly due to the consciousness that in any case some clever pruning should be applied to the skeleton before its use, to get rid of branches not significant in the problem domain. Thus, the task of removing superfluous pixels, and of obtaining a skeleton structure with an attractive shape is committed to an appropriate pruning phase (for instance, modeled on the one used in [1]).

To find the ridge points having unit distance from B, the operators of Figure 4 have been used, instead of those devised for all the remaining pixels in EDM. Otherwise, the path growing process could be not effective to connect the skeleton. In fact, if the operators of Figure 3 were used, one could find components of ridge points labeled '1' (from which paths are expected to be started) having no neighbors with higher label. Thus, we need to augment the components of ridge points labeled '1', by marking in EDM further pixels labeled '1', to force adjacency to pixels with higher label. This goal is reached by using the operators of Figure 4, which in [2] have been shown to identify connected components 8-adjacent to pixels inner in the object (in our case, having higher label). Two examples are shown in Figure 7, where the black dots mark the pixels detected as ridge points by the operators of Figure 3, and the white dots are the additional pixels necessary to start the path.

Geometric configurations including equilabeled pixels, as the ones indicated by (black and white) dots in Figure 7, cannot be present inside EDM. To this purpose, consider the Cartesian coordinates system centered on the pixel p, as shown in Figure 8. Let w be the pixel of B nearest to p. If w is not on a coordinate axis, its distance from at least one of the pixels denoted by E is smaller than its distance from p, which accordingly cannot share the same label with the E neighbors. Contradiction. If w is located on a coordinate axis, the only case its position does not prevent p to be labeled as the E neighbors, occurs when w coincides with a dashed pixel, i.e., when p has unit distance from B.

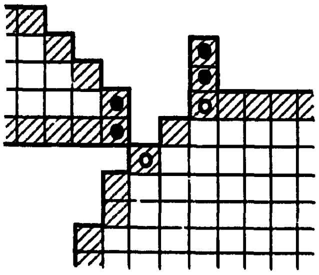

Figure 7. Portion of an object. Dashed pixels have unit distance from the complement of the object. Black dots mark pixels identified as ridge points by the operators of Figure 3. White dots should also be detected as ridge points to allow skeleton connectedness through the path growing process.

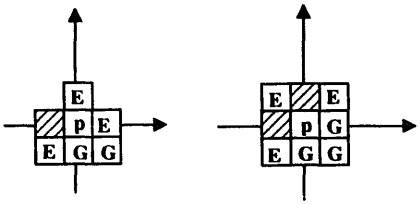

Figure 8. The only possibility for p to be labeled as its neighbors denoted by E, is that p is labeled '1'.

Extensive experimental work has been carried out, which did not show any failure of the algorithm. However, the formal proof of the correctness of the algorithm is not given here, since the treatment of such a topic is beyond the scope of this article. Indeed, one should show that the obtained skeleton is connected, has the same connectivity order of the object, and every loop of the skeleton surrounds a hole of the object. The proof could be modeled on similar proofs regarding skeletonization processes performed on other distance maps, e.g., [1]. In fact, even if computed according to different distance functions, the obtained maps are similarly constrained. Moreover, to call skeleton the obtained set, also the reconstructability property should be satisfied. This means that the local maxima of EDM should all be included in the set of the ridge points. Research on local maxima detection is in progress [6].

References

[1] Arcelli, C. and G. Sanniti di Baja (1985). A width-independent fast thinning algorithm. *IEEE Trans. Pattern Anal. Machine Intell.* 7, 463-474.

[2] Arcelli, C. and G. Sanniti di Baja (1987). A contour characterization for multiply connected figures. *Pattern Recognition Letters* 6, 245-249.

[3] Arcelli, C. and G. Sanniti di Baja (1989). A one-pass two-operations process to detect the skeletal pixels on the 4-distance transform. *IEEE Trans. Pattern Anal. Machine Intell.* 11, 411-414.

[4] Blum, H. (1973). Biological shape and visual science. *J. Theor. Biol.* 38, 205-287.

[5] Borgefors, G. (1986). Distance transformation in digital images. *Computer Vision, Graphics, and Image Processing* 34, 344-371.

[6] Borgefors, G., I. Ragnemalm and G. Sanniti di Baja (1991). The Euclidean distance transform: finding the local maxima and reconstructing the shape. In preparation.

[7] Danielsson, P.E. (1980). Euclidean distance mapping. *Computer Graphics and Image Processing* 14, 227-248.

[8] Dorst, L. (1986). Pseudo-Euclidean skeletons. *Proc. 8th ICPR*, Paris, 286-288.

[9] Klein, F. and O. Kubler (1987). Euclidean distance transformations and model-guided image interpretation. *Pattern Recognition Letters* 5, 19-29.

[10] Kubler, O., F. Klein, R. Ogniewicz and U. Kienholz (1990). Isolation and identification of abutting and overlapping objects in binary images. In: V. Cantoni, L.P. Cordella, S. Levialdi and G. Sanniti di Baja, Eds., *Progress in Image Analysis and Processing*. World Scientific, 340-347.

[11] Ragnemalm, I. (1989). The Euclidean distance transform and its implementation on SIMD architectures. *Proc. 6th Scandinavian Conf. on Image Analysis*, Oulu, Finland, 379-384.

[12] Ragnemalm, I. (1990). Generation of Euclidean Distance Maps. Thesis No. 206, Dept. Electrical Engineering, Linkoping University.

[13] Shih, F.Y. and C.C. Pu (1990). Medial axis transformation with single-pixel and connectivity preservation using Euclidean distance computation. *Proc. 10th ICPR*, Atlantic City, 723-725.

[14] Yamada, H. (1984). Complete Euclidean distance transformation by parallel operation. *Proc. 7th ICPR*, Montreal, 69-71.