FAST EUCLIDEAN MORPHOLOGICAL OPERATORS USING LOCAL DISTANCE TRANSFORMATION BY PROPAGATION, AND APPLICATIONS.

O Cuisenaire and B Macq.

Communications and Remote Sensing Laboratory, Université catholique de Louvain, Belgium.

We propose a new method to compute the morphological dilation of a binary image with a circular structuring element of any given size, on a discrete lattice.

The algorithm is equivalent to applying a threshold on an exact Euclidean distance map, but computations are restricted to a minimum number of pixels. The complexity of this dilation algorithm is compared to the complexity of the commonly used approximation of circular structuring elements and found to have a similar cost, while providing better results.

INTRODUCTION

Dilation and erosion are the basic operators of mathematical morphology (see Serra (2)). The dilation of a set of points X by a structural element B is written $X \oplus B$ and is defined as follows.

$$X \oplus B = \left\{ x + b \big| (x \in X) \wedge (b \in B) \right\} \qquad (1)$$

Erosion is the dual of dilation, i.e. the complement of a dilation performed on the complement set of X. Other morphological operators can be derived by combining dilation and erosion, and provide a full toolkit of operators to process objects in a binary image according to their shape.

Symmetrical and circular structural elements (SE) play a central role in mathematical morphology in the continuous plane, since they provide an isotropic treatment of the image. On the other hand, for digital images, circular SE are rarely used because there is no simple and efficient implementation of the dilation by such a SE on a discrete lattice. Indeed, a direct application of the definition above leads to a computation cost of complexity $o(n^2.d^2)$ for an image of size $n \times n$ and a SE of radius d. Such a cost is prohibitive for many pattern recognition applications.

In section 2, we review a variety of techniques used to obtain fast morphological operators with structural elements B that are approximately or perfectly circular. In section 3, we propose a new algorithm that performs the dilation by a circular element based on exact Euclidean distance transformation by propagation. In section 4 we assess its computational complexity and memory requirements, and compare it to the approximations of section 2. Finally, in section 5 we present an application from histology where a family of dilations with circular structural elements of all sizes between 0 and d are needed.

FAST MORPHOLOGICAL OPERATORS

Fast implementations of the dilation operator rely on a number of properties of this operator.

Decomposition of the structural element

Dilation is an associative operation, i.e.

$$X \oplus \left(B \oplus B' \right) = (X \oplus B) \oplus B' \qquad (2)$$

Some structural elements can be decomposed into simpler elements, as illustrated at **Figure 1**. Applying the dilations with the smaller elements iteratively instead of the large SE at once reduces the complexity of the dilation. For a square SE of radius d, it goes down from $o(n^2.(2d+1)^2)$ to $o(n^2.2(2d+1))$. The dilation by a diamond SE is performed in $o(n^2.4d)$.

This can be further improved for the square SE. Indeed, one dimensional dilations can be implemented in $o(n^2)$, and therefore the square SE dilation are $o(2n^2)$.
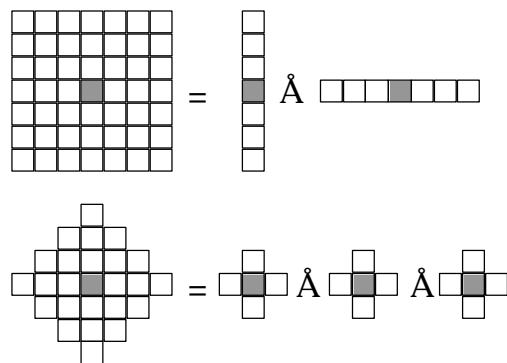

Figure 1 : Decomposition of the square and diamond structural elements into 2 and d elementary structural elements respectively. For the sake of the illustration, we have chosen d=3.

These two SE are very poor approximation of a circle. A common improvement is to use a combination of both SE, which leads to an octagonal SE. An hexagonal SE can be produced by the same composition principle when working on an hexagonal grid. All those implementations have a $o(n^2.d)$ complexity.

CONTOUR PROCESSING

To compute a dilation of a set X, only the edge of this set needs to be considered. More accurately, we have

$$X \oplus B = X \, Y(\partial(X) \oplus B) \qquad (3)$$

where $d(X)$ is the edge of X, i.e. the set of pixels of X with at least a direct neighbor not belonging to X. If l is the length of the contour of X, i.e. the cardinal of set $d(X)$, then the computational complexity is reduced to $o(l.d^2)$ for any SE of radius d, plus a small $o(n^2)$ term to determine the pixels belonging to $d(X)$.

By combining the properties of equations (2) and (3), we have the basis for contour-processing algorithms for decomposable SE as in Van Vliet and Verweer (5), whose complexity is further reduced to $o(l_{max}.d)$ where l_{max} is the maximal size of the contour during the iterations with elementary SE..

Finally, Vincent (7) proposes an algorithm using both contours $d(X)$ and $d(B)$, of the set X and the structural element. This algorithm has a complexity proportional to the product of the number of pixels in each contour, which means $o(l.d)$ for a circular element of size d. It can also be expressed as $o(A)$ where A is the cardinal of $(X\text{Å}B)\backslash X$.

Threshold of a distance transformation

When B is a ball, i.e. when it is defined as

$$B = \left\{ b \middle| d(b,0) \le d \right\} \qquad (4)$$

then, dilation by B can also be expressed as the threshold of a distance function, i.e.

$$X \oplus B = \left\{ x \middle| d(x, X) \le d \right\} \qquad (5)$$

where d(x,S) is the distance from pixel x to the set S, i.e. the distance from pixel x to the nearest pixel belonging to S, as defined in Borgerfors (3). This means the dilation can be considered as a threshold of a distance transformation (DT).

The square and diamond SE can be considered as balls for distances defined using the chessboard and city-block metrics respectively. Better approximations of the circle can be obtained using Chamfer metrics. The corresponding DT algorithms are all of a $o(n^2)$ complexity, and so are the dilations. In general, this is less efficient than contour processing techniques, although this is image dependant.

Use of Euclidean DT

Ragnelmam (8) proposed to combine contour processing and DT thresholding. He merged the bucket sorting propagation concept of Piper and Granum (4) and Verwer et al. (6) with the quasi Euclidean distance metric of Danielsson (1). The resulting quasi Euclidean DT by propagation can be restricted to the pixels with a distance smaller than d, a very efficient implementation which yields a complexity similar to that of other contour-processing methods. It can be expressed as $o(A)$ where A is the cardinal of $(X\text{Å}B)\backslash X$.

Unfortunately, as Ragnelmam (9) points out, the DT of Danielsson is not exactly Euclidean and can lead to small errors in particular object pixel configurations, as illustrated at **Figure 2**. While this is of small practical importance for a simple dilation, it can have catastrophic consequences when one considers other operators such as a morphological closing, defined as a dilation followed by an erosion, or

$$X \bullet B = ((X \oplus B)^c \oplus B)^c \qquad (6)$$

using only the dilation and set complementation.

Considering the example of **Figure 2**, a set $X=\{a,b,c\}$ and a structural element $B = \{b \mid d(b,0) \le 13\}$, pixel x is wrongly classified as belonging to $(X \oplus B)^c$. During the second dilation, the error is not reproduced since the configuration of pixels of $\partial((X \oplus B)^c)$ is not problematic. Therefore, pixel b is included in $(X \oplus B)^c \oplus B$, and the morphological closing of X by B does not contain b, i.e. $X \bullet B = \{a,c\}$. This contradicts a fundamental property of the morphological closing: extensivity, i.e.

$$X \subseteq (X \bullet B) \qquad (7)$$

It is possible to avoid this problem by using one of the exact Euclidean DT algorithms by propagation later proposed by Ragnelmam (9) or Eggers (10). Unfortunately those algorithms have a $o(n^3)$ complexity for $n \times n$ images in the worst case scenario. This means a $o(l.d^2)$ complexity for propagation limited to distance d. This is not asymptotically better than the direct application of equation 3 with an arbitrary structural elements of radius d.
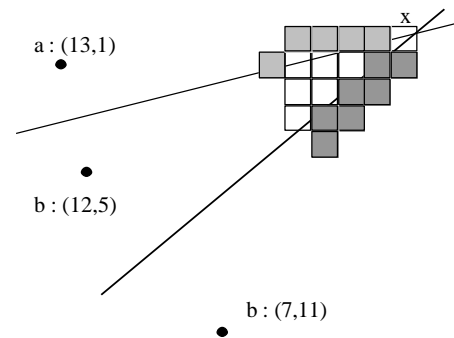

Figure 2 : Because the tiles of digital Voronoi divisions are not connected sets, pixel X will be mislabeled by the distance transformations of (1) and (8). X is closer to *b*

with $d(x,b)=13$ than to *a* or *c* with $d(x,a)=d(x,c)=\sqrt{170}$ but all his neighbors are closer to a or c than b. Hence it is not reached by the propagation and leads to an error in the DT from the set of pixels {a,b,c}

EUCLIDEAN OPERATOR BY EXACT EDT PROPAGATION.

In Cuisenaire and Macq (13), we propose a new exact Euclidean distance transformation by propagation. It works in two steps. First, a quasi Euclidean map is computed using ordered propagation through bucket-sorting of the pixels in the propagation front (9), with the 4SED neighborhood of (1). Secondly, the map is corrected by further propagating the limited number of pixels that failed to propagate at step 1. This further propagation is restricted within directional neighborhoods (13) of a size depending on the distance (see Table 1).

We use 2 lists of buckets to store the pixels of the propagation front. For each pixel, we remember its location p and his relative position *dp* to the nearest pixel of set *X*. (*p,dp*) is stored in the bucket labeled with the square of the Euclidean distance, i.e. $d^2(dp) = dp_x^2 + dp_y^2$.

The algorithm to produce the map *D* of square distances thresholded at d^2 is written:

```
for all pixel p ∈ X
   if (p+n) ∉ X for any n ∈ { (0,1),(0,-1),(-1,0),(1,0)}
     put (p,(0,0)) in bucket1(0)
     D(p) = 0
   else D(p) = d²+1
i = 0
while i < d²
   for all ( p , dp ) in bucket1( i )
     for all n ∈ { (0,1),(0,-1),(-1,0),(1,0)}
       if d² (dp+n) < D (p+n)
         D (p+n) = d² (dp+n)
         put ( p+n , dp+n ) in bucket1 ( d² (dp+n) )
     if p was not propagated for any n
       put ( p , dp ) in bucket2( i )
   free bucket1(i)
   i = i+1
i = 0
while i < d²
   for all ( p , dp ) in bucket2( i )
     for all n ∈ directed neighbor N(dp)
       if d² (dp+n) < D (p+n)
         D (p+n) = d² (dp+n)
         put ( p+n , dp+n ) in bucket2 ( d² (dp+n) )
   free bucket2(i)
   i = i+1
```

where the directed neighbor *N(dp)* for pixel *p* with *dp* in the first quadrant is made of all pairs (*i,j*) with

$$(i+1).\frac{dp_x}{dp_y} \le j \le 1+i.\frac{dp_x}{dp_y}$$

$$0 \le i \le i_{max}(d^2(dp))$$

(8)

with $i_{max}(d^2)$ found in Table 1. When *dp* is in the other 3 quadrants, signs are changed accordingly. The correctness of this algorithm is proved in (13). All pixels *p* such as $D(p) \le d^2$ belong to $X \oplus B$ with *B* a ball of size d.

TABLE 1 : $i_{max}(d^2)$, size of the neighborhood needed for a pixel at distance d.

d²	i_max
$0 \to 1$	0
$2 \to 115$	1
$116 \to 519$	2
$520 \to 2016$	3
$2017 \to 4609$	4
$4610 \to 10599$	5
$10600 \to 18751$	6
$18752 \to 34216$	7
$34217 \to 52881$	8

COMPLEXITY AND MEMORY REQUIREMENTS

As pointed out in (13) and (12), the propagation with multiple neighborhood algorithm for Euclidean DT has a $o(n^2)$ complexity for $n \times n$ images. Because there are in general few non-propagating points and because the directed neighborhoods are kept small, the additional computational cost of step 2 is usually kept a fraction of the time needed for step 1. Even in the worst-case scenario, step 2 is no more than $o(n^2)$. This means that the computation time per pixel is a constant.

With distance propagation restricted to $D(p) \le d^2$, the computation cost is $o(A)$ where *A* is the total number of pixels involved in the propagation, i.e. the cardinal of $(X \oplus B)\backslash X$. This cost is similar to the cost of contour processing techniques and faster than all methods based on the chamfer distance transformation, while our algorithm provides a truly circular structural element.

The price for this computational efficiency is the additional memory requirements. While all other methods, including Ragnelmam's, can work on binary images, we need to store the distance map D explicitly. This requires to work on 8 bit images for $d \le 8$, 16 bits for $d \le 256$, ... Also, the buckets need to be stored, which requires d^2 dynamic lists. Fortunately, the same memory locations can be used for the bucket1 and bucket2 structures, since bucket1(*j*) is empty for all *j<i* and bucket2(*j*) is empty for *j>i*.

A further improvement can be obtained by merging the two steps into one. This way, only one bucket structure is used, and we can perform the threshold dynamically when each point is considered in the propagation. Furthermore, only a small portion of the buckets contain pixels at a given time, those corresponding to the current distances of the propagation front. Therefore, we

can reuse the buckets periodically, with a minimal distance between reused buckets of

$$M = (d + i_{max}(d^2).\sqrt{2})^2 - d^2 \approx 3.d.i_{max}(d^2). \qquad (9)$$

The algorithm is then written

```
for all pixel p ∈ X
   if (p+n) ∉ X for any n ∈ { (0,1),(0,-1),(-1,0),(1,0)}
      put (p,(0,0)) in bucket (0)
      D(p) = 0
   else D(p) = d²+1
i = 0
while i < d²
   for all ( p , dp ) in bucket( i mod M)
      D(p) = 0
      for all n ∈ { (0,1),(0,-1),(-1,0),(1,0)}
         if d² (dp+n) < D (p+n)
            D (p+n) = d² (dp+n)
            put ( p+n, dp+n ) in bucket ( d² (dp+n) mod M)
      if p was not propagated
         for all n ∈ directed neighbor N(dp)
            if d² (dp+n) < D (p+n)
               D (p+n) = d² (dp+n)
               put ( p+n, dp+n ) in bucket(d²(dp+n) mod M)
   free bucket( i mod M)
   i = i+1
```

The result of this algorithm is an image D with $D(p)=0$ for all pixels of $X \oplus B$ and $D(p)=d^2+1$ for all others.

APPLICATION

In Cuisenaire et al. (11), one tries to detect neuronal fibers in microscopic images (**Figure 3**a). Those fibers have an approximately circular bright center surrounded by a dark myelin sheet of constant thickness.

After identifying axon candidates (areas 1-6) and discarding other areas (areas x,y) in the thresholded and simplified image of **Figure 3**b, a major issue is how to separate the aggregate fibers, i.e. how to split the pixels in the black area into myelin sheets around each axon.

The thickness of the myelin sheet around one axon is evaluated as follows: first, we define X_d the set of pixels at distance d of a set X of pixels as

$$X_d = (X \oplus B_d) \setminus (X \oplus B_{d-1}) \qquad (10)$$

with B_d a ball of size d. The thickness of the myelin sheet is then the smallest distance d for which X_d contains more white than black pixels.

This can very efficiently implemented by modifying the stopping criterion in the above algorithm as follows

```
Same initialization
for all i
   white(i) = black(i) = 0
nblack = 0; nwhite = 0; i = 0;
while nblack ≥ nwhite
   for all ( p , dp ) in bucket( i mod M)
      if D(p) ≠ 0
         D(p) = 0
         if p is white
            white(i) = white(i) +1
         else black(i) = black(i) + 1
      for all n Î { (0,1),(0,-1),(-1,0),(1,0)}
      ...
   free bucket ( i mod M)
   nblack = nblack + black(i)
   nwhite = nwhite + white(i)
   for k = ( sqrt(i-1) - 1 )² to (sqrt(i) - 1 )²
      nblack = nblack – black(k)
      nwhite = nwhite – white(k)
   i = i+1
```

where the core of the algorithm (in italic) is left unchanged. The variables *nblack* and *nwhite* are the number of black and white pixels in X_d. They are computed from the variables *white(i)* and *black(i)* , the number of black and white pixels at distance $d^2=i$.

Instead of a fixed ending criterion ($i<d^2$), the dynamically computed criterion (*nblack $\geq$ nwhite*) is used. Obviously, this requires a negligible additional cost, while the direct application of the definition of X_d would require the computation of the dilation of X by all elements of radius varying from 0 to d, a o(l.d²) problem.

This algorithm is applied for all axons candidates sharing the same neighboring black area, in order of decreasing area. The myelin sheet around each axon is found as $X \oplus B_d \setminus X$, as illustrated at **Figure 3**d.
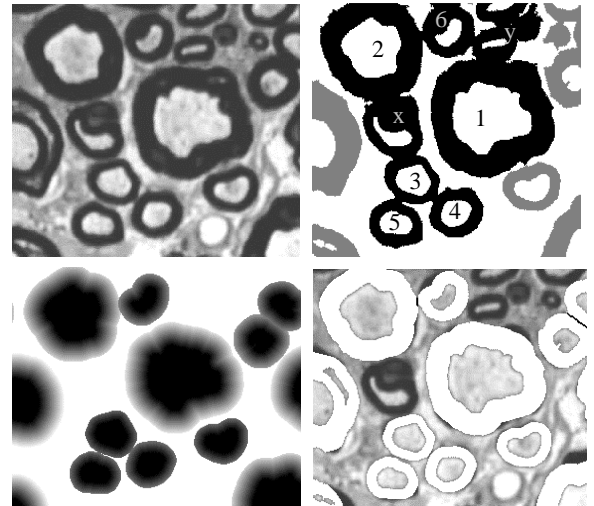

Figure 3 : Neuronal fibers separation. Top left (a): original image. Top right (b): simplified binary image. Bottom left (c): distance map. Bottom right (d): detected myelin sheets overlaid on the original image.

DISCUSSION

With a computational complexity proportional to the size of $(X \oplus B) \setminus X,$ our algorithm performs as well as, but not significantly better than (7). Both algorithms outperform any other method, either in precision or in cost.

Both algorithms have additional capabilities that can make them more interesting for a particular application: On one hand, Vincent's algorithm can also be used to perform dilations with structural elements of arbitrary shape. On the other hand, our algorithm is limited to Euclidean balls, but can be stopped at any distance d and provide $X \oplus B$ for a ball B of size d. For instance, it can easily perform dilations with elements B of increasing size, until some criterion is met.

CONCLUSION

Contour-processing algorithms and thresholding of distance transformations are the two most efficient implementations of morphological dilations but usually require the use of simple structural elements such as squares or diamonds.

By combining the two techniques and using a o(n²) exact Euclidean DT by propagation (13), we implement Euclidean mathematical morphology operators with a computation time and complexity similar to its usual approximations, i.e. o(A) where A is the number of pixels involved in the propagation, i.e. the cardinal of $(X \oplus B) \setminus X$.

ACKNOWLEDGEMENTS

REFERENCES

1. Danielsson P E, 1980, "Euclidean distance mapping". CGIP, 14, 227-248.

2. Serra J, 1982, "Image analysis and mathematical morphology", Academic Press, London, UK.

3. Borgefors G, 1986 "Distance transformations in digital images". CVGIP, 34, 344-371

4. Piper J and Granum E, 1987, "Computing distance transformations in convex and non-convex domains", Pattern Recognition, 20, 599-615.

5. Van Vliet LJ and Verwer BJH, 1988, "A contour-processing method for fast binary operations", Pattern Recognition Letters, 7, 27-36.

6. Verwer BJH, Verbeek PW and Dekker ST, 1989, "An efficient uniform cost algorithm applied to distance transforms". IEEE trans. PAMI, 11, 425-429.

7. Vincent L, 1991, "Morphological transformations of binary images with arbitrary structuring elements", Signal Processing, 22, 3-23

8. Ragnelmam I, 1992, "Fast erosion and dilation by contour processing and thresholding of distance maps", Pattern Recognition Letters, 13, 161-166

9. Ragnelmam I, 1992, "Neighborhoods for distance transformations using ordered propagation" CVGIP - Image Understanding, 56, 399-409

10. Eggers H, 1998, "Euclidean distance transformations in Z² based on sufficient propagation", CVIU, 69, 106-116.

11. Cuisenaire O, Romero E, Veraart C and Macq B, 1999, "Automatic detection and measurement of axons in microscopic images", Proceedings SPIE Medical Imaging 1999, San Diego (CA), February 20-26th.

12. Cuisenaire O and Macq B, 1999, "Fast and exact signed Euclidean distance transformation with linear complexity", Proceedings of ICASSP99, Phoenix (AR), March 15-19th.

13. Cuisenaire O and Macq B, "Fast Euclidean distance transformation by propagation using multiple neighborhoods", submitted to CVIU.