ELSEVIER

Image and Vision Computing 15 (1997) 845–854

image AND vision COMPUTING

Optimal 3 × 3 decomposable disks for morphological transformations

Maria Vanrell*, Jordi Vitrià

Computer Vision Center, Universitat Autònoma de Barcelona, Edifici O, 08193 Bellaterra, Barcelona, Spain

Received 1 August 1996; revised 4 March 1997; accepted 6 March 1997

Abstract

Morphological operations can be efficiently computed on parallel architectures using the decomposition of the structuring elements. In some cases, decomposition is guided to optimize computation for a given underlying hardware, but in other cases it is the shape of the structuring elements which directs the decomposition. In this paper we present a method to decompose disks. Morphological operations with isotropic structuring elements present interesting properties as shape and size descriptors. The method developed is based on a constraint-satisfaction algorithm that gives an optimal decomposable disk. Optimality is given by the shape of the disk since it is the best discrete approximation of a circle that allows a 3 × 3 decomposition. © 1997 Elsevier Science B.V.

Keywords: Mathematical morphology; Structuring element; Disk decomposition; Discrete disk approximation

1. Introduction

Mathematical morphology is a theory which has become an important nonlinear approach to image analysis [1,2]. It has been successfully applied due to its ability to deal with intuitive notions such as shape, size or connectivity. Another important reason for its success is the simplicity of being computationally implemented. Most of its operators have an intuitive interpretation about how they transform the image. In general, morphological transformations change the gray level of the image depending on the given structuring element. The shape of the element usually determines the image transformation result.

Parallel computers have been widely used to improve morphological algorithms. Also, some specialized architectures, such as the Cytocomputer, have been built to perform effective morphological operations [3]. To exploit this hardware, it is important to compute erosions and dilations as recursive processes. Thus, the decomposition of the structuring elements is an essential step in this process.

The problem of decomposing structuring elements has been studied in depth in previous works [4,5]. In some cases the decomposition has been guided by the shape of the structuring element. The work of Xu [6] considers the decomposition problem for convex structuring elements. A very interesting approach, developed by Park and Chin [7], decomposes structuring elements of arbitrary shape.

In this paper, we present a method to obtain discrete circular structuring elements and their decomposition. The isotropy property of the structuring elements is very important in image analysis, especially for shape and size descriptors. Pattern spectrum [8] for shape description is based on a family of morphological openings with circular structuring elements.

Isotropic morphological openings have been used to compute granulometries which provide size distribution functions of nodules in radiographic images [9].

In some computer vision approaches, image is represented by a scale space image constructed by an isotropic smoothing of the image for different scales. The vision problems have shown the impracticability of using linear processes alone. In this sense, the morphological operators are an interesting nonlinear approach to construct a nonlinear scale space [10].

For a computational model of preattentive texture perception, isotropic structuring elements have been used to compute nonlinear postinhibition responses [11,12], where the morphological dilation at different sizes allows best-tuned responses to be extended over an isotropic neighborhood. The introduction of morphological operations has allowed a fast algorithm of the model to be constructed.

Also, we can state that for nonlinear filtering gray-level morphology is an important technique. Isotropic structuring elements can have an important role in preserving the shape of the image objects in a filtering step.

Finally, we want to consider that in practical vision

* Corresponding author.

PII S0262-8856(97)00026-7

applications time requirements have usually implied solutions based on binary morphological operations on the binarized image versions, given that there exist efficient algorithms to compute them based on the image distance map [13]. Sometimes, success in these applications could be improved by computing morphological operations directly on gray level images. Reducing costs for gray-scale flat morphology can be an interesting contribution in this field.

To sum up, in this work we present a method to obtain the shape of isotropic structuring elements and to compute their decomposition in 3×3 basic structuring elements. In order to do this, we have organized the paper in three parts. First, we give a brief introduction and basic definitions to the decomposition problem. Second, we define the shape of a decomposable disk. Finally, we give the decomposition of the defined disk.

2. Background and definitions

In this section we introduce a brief review of recently developed methods to decompose structuring elements from their boundary representation. For this purpose we give the concept of convex and concave boundaries, expressed in terms of a chain code representation, as well as some propositions which define the decomposability of a given element.

In order to improve the morphological erosion and dilation computation, we can use the following properties:

$$f \ominus B = \{[\ldots((f \ominus B^1) \ominus B^2)\ldots] \ominus B^n\} \tag{1}$$

$$f \oplus B = \{[\ldots((f \oplus B^1) \oplus B^2)\ldots] \oplus B^n\} \tag{2}$$

where $B = B^1 \oplus B^2 \oplus \ldots \oplus B^n$ represents the decomposition of B. Therefore, we can compute a basic morphological operation by applying the operation recursively on the decomposition of the original structuring element. The decomposition allows to reduce the number of basic operations, and makes the region of support smaller.

The optimality of a given decomposition depends on the underlying image processing hardware. On classical sequential architectures, the morphological operation depends on the number of shifts, that is, the number of points that appear in all the decomposed elements. Furthermore, on general array processors the cost usually depends on the required number of shifts. However, for 4-connected array processors the cost is the sum of the distances of all pixels in the structuring elements in a city block metric. Finally, on pipeline machines the region of support of the structuring elements is fixed and the cost depends on the number of basic structuring elements of given dimensions.

In this work we do not pretend to obtain an optimal decomposition of disks for an especial underlying architecture. The research is aimed at constructing optimal decomposable disks. For that purpose we will consider decomposition with 3×3 basic structuring elements and will regard the number of required shifts for a given decomposition as a cost evaluator.

We will base our approach on [7]. This work gives the method of obtaining the decomposition of simply connected binary structuring elements of arbitrary shape into 3×3 elements. The decomposition is obtained from the boundary of the structuring element, which is represented by the Freeman's chain code. The boundary chain code contains sufficient shape information to obtain the decomposition.

In order to be able to decompose structuring elements of arbitrary shape we have to distinguish two types of boundaries: *convex* and *concave* boundaries, which we are now going to define.

Definition 2.1. A binary structuring element S is convex if it is an intersection of discrete half-planes in the direction of multiples of 45°. The chain code of the convex boundary of a given element is represented by

$$S = 0^{x_0}1^{x_1}2^{x_2}3^{x_3}4^{x_4}5^{x_5}6^{x_6}7^{x_7} = [i^{x_i}]_{i=0,\ldots,7} \tag{3}$$

where $x_i > 0$ means that the chain code i repeats x_i times and $x_i = 0$ when the direction i does not appear at the boundary. In order to be a valid chain code for a convex structuring element it must satisfy the following conditions:

$$x_7 + x_0 + x_1 = x_3 + x_4 + x_5 \tag{4}$$

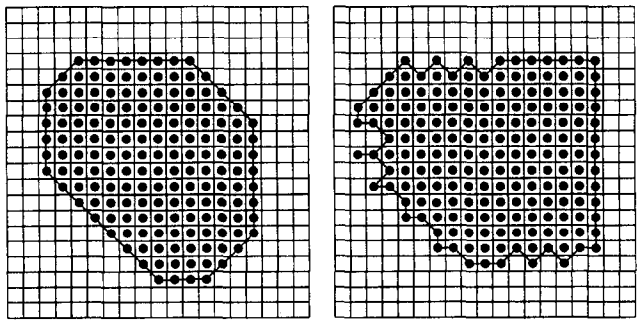
$$x_1 + x_2 + x_3 = x_5 + x_6 + x_7 \tag{5}$$

We will also represent the chain code S as the array $(x_0,x_1,x_2,x_3,x_4,x_5,x_6,x_7)$.

In Fig. 1(a) we can see a convex structuring element.

A structuring element contains a concave boundary if the shape of the element cannot be expressed as Eq. (3). Hence, there exists an infinite number of different concave boundaries.

Considering that we are defining the shape of a structuring element in order to obtain its decomposition, we can restrict the set of concave boundaries that we are going to deal with. The constraint is given by the following fact: any



(a) (b)

Fig. 1. (a) Convex structuring element with chain code $0^3 1^3 2^7 3^4 4^7 5^2 6^5 7^7$; (b) concave structuring element with chain code $Q_{U0}Q_{J0}Q_{L0}Q_{r0}0^2 Q_{Vi}^2 Q_{r1} 2^{12} 4^6 Q_{V5}^3 5^3 6$.

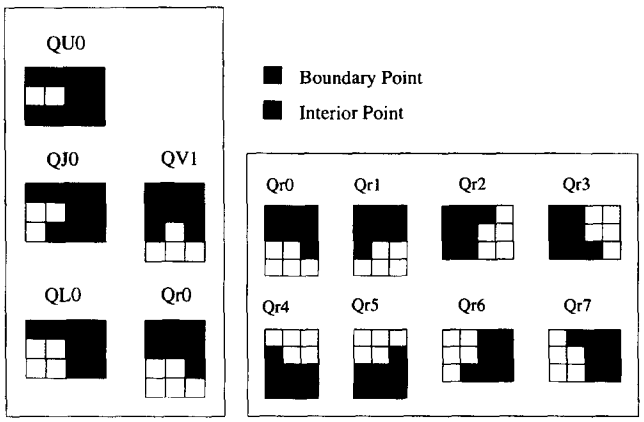


Fig. 2. (a) Five types of concave boundaries of 3 × 3 dimensions, denoted as U, J, L, V and r; (b) concave boundaries of type r in the eight possible starting directions, denoted by the number i in Q_{ri}.

concave boundaries of the initial structuring element must be contained in the basic decomposed elements.

The above statement takes us to the consideration that, for a 3 × 3 decomposition, we can construct the set of all concave boundaries that can be contained in a 3 × 3 element. In Fig. 23(a) we can see the five possible types of boundaries,3 which are denoted as U, J, L, V and r. We have also to consider all their possible rotations. Then, any specific 3 × 3 boundary is denoted as Q_{Ti} where T indicates the type and i denotes the starting chain code direction.

The constraint introduced by the dimensions of the basic structuring element allows us to make a restricted definition of the boundaries of a concave structuring element. Its boundary is formed by non-overlapping concave and convex boundaries.

Definition 2.2. A binary and connected structuring element S is concave if its boundaries can be represented by the following expression

$$S = \left[Q_{T0}^{S_{T0}}\right]_{T=U,J,L,r} 0^{S_0} \left[Q_{T1}^{S_{T1}}\right]_{T=J,V,r} 1^{S_1} \ldots \left[Q_{T7}^{S_{T7}}\right]_{T=U,J,L,r} 7^{S_7} \quad (6)$$

where S_i denotes the repetition of the i direction in convex boundaries, and S_{Ti} denotes the repetition of the Q_{Ti} concave boundary. We have to assume that the chain code superscripts have to satisfy some constraints in order to represent the shape of a valid chain code.

From this definition we can deduce that if $S_{Ti} = 0 \forall Ti$, then X is convex; otherwise, we have a concave structuring element with boundaries that can be contained by basic 3 × 3 elements allowing decomposition.

Before proceeding with the decomposition problem, we give one more definition.

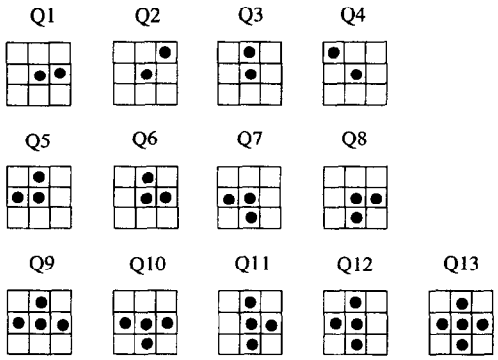


Fig. 3. Set of 3 × 3 prime factors to decompose convex structuring elements.

Definition 2.3. A given element A is a factor of S if there exists a structuring element B such that $S = A \oplus B$. A factor A of S is a *prime factor* iff every factor of A is equivalent to A or is a one pixel image.

Now, we are going to introduce some decomposability criteria for convex and concave structuring elements.

The problem has been solved for convex structuring elements in [6]. Its results can be summarized in the following proposition.

Proposition 2.1. Every convex structuring element S is decomposable into a sequence of dilations by a set of prime factors $Q1,3Q_2,\ldots,Q_{13}$, given in Fig. 3.

A constructive proof of this proposition provides an elegant algorithm presented in [6], which has been improved in [14] for 4-connected array processors. This algorithm has been included as the last step in a more general method to decompose the concave structuring element [7].

In the same way that 3 × 3 concave boundaries have been defined, the 3 × 3 concave prime factors can also be constructed by considering all possible combinations of concave boundaries in 3 × 3 structuring elements. We will denote this set of prime factors as $\{A_i\}$. A subset of all these prime factors can be seen in Fig. 8. The set $\{A_i\}$ of factors has been the basis for constructing a general decomposition algorithm in [7]. Before giving the general proposition determining decomposability of any structuring element, we are going to define some notation.

Let S be the concave element that we want to decompose, and $\{A_i\}$ be the set of all prime factors that have 3 × 3 concave boundaries. Hence, if S is decomposable then

$$S = C_1 \oplus C_2 \oplus \ldots \oplus C_m \oplus B$$

where $C_j \in \{A_i\}$ and B is a convex factor of S and consequently decomposable by Proposition 2.1.

By definition of the concave boundary we can state that the boundary of S is formed by

1. m distinct concave boundaries, $V_1,V_2,\ldots,V_m$, where V_k is one of Q_{Ti}'s where $S_{Ti} \neq 0$;

2. l distinct convex boundaries, $d_1,d_2,\ldots,d_l$, where d_k is one of i's directions where $S_i \neq 0$.

From these boundaries we can take n concave prime factors from $\{A_i\}$, that is, $A_1,\ldots,A_n$ that have boundaries in common with S, and consequently can be prime factors of S.

The above notation allows us to construct the matrices $\boldsymbol{\Theta}$ and $\boldsymbol{\Omega}$, of dimensions $m \times n$ and $l \times n$, respectively:

$$[\boldsymbol{\Theta}]_{ij} = \text{number of } V_i \text{ in } A_j$$

$$[\boldsymbol{\Omega}]_{ij} = \text{number of } d_i \text{ in } A_j$$

Moreover, we construct $\boldsymbol{Y}$ and $\boldsymbol{Z}$ vectors of m and l dimensions respectively, from the chain code of the element, S:

$$[\boldsymbol{Y}]_i = \text{number of } V_i \text{ in } S$$

$$[\boldsymbol{Z}]_i = \text{number of } d_i \text{ in } S$$

Finally, we define a vector $\boldsymbol{X}$ of m variables with non-negative integer values that will represent the decomposition of S. The decomposition of S will be derived from the following proposition.

Proposition 2.2. A given concave structuring element S is decomposable if there exists an X such that

1. $\boldsymbol{\Theta X} = \boldsymbol{Y}$
2. $\boldsymbol{\Omega X} \leq \boldsymbol{Z}$
3. $x_1A_1 \oplus \ldots \oplus x_nA_n \oplus B$ *is simply connected, and* x_iA_i *denotes* x_i *dilations by* A_i
4. B *is a convex factor of S, defined by the chain code* $(b_0,b_1,\ldots,b_7)$, *and* $b_i = [Z - \boldsymbol{\Omega X}]_i, \forall i : 0 \leq i \leq 7$. *Therefore, if such an* $\boldsymbol{X}$ *exists, the decomposition of S is*

$$S = x_1A_1 \oplus \ldots \oplus x_nA_n \oplus B$$

The definitions and propositions given above enable us to compute the 3×3 decomposition of any structuring element. This fact allows us to develop a general method to decompose circular structuring elements.

The problem now consists of finding the chain code for a good approximation of a circle whose decomposability can be assured. Towards this objective, we firstly will define a decomposable chain code that can approximate a disk. Secondly, we have to develop a fast algorithm to find this disk for any given radius. Finally, we will use Proposition 2.2 to obtain the general decomposition and then we will discuss the results.

3. A method for disk decomposition

As we have argued before, morphological operations with circular structuring elements are interesting for different applications. Therefore, the decomposition of such elements plays an important role towards the objective of optimizing algorithms.

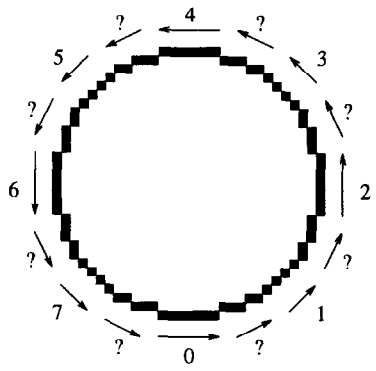


Fig. 4. A non-decomposable disk in 3×3 basic structuring elements.

Discrete approximation of circles has been widely studied in the field of computer graphics [15]. The most efficient algorithms are incremental circle generators that apply the midpoint criterion. That is, one given coordinate is increased step by step, and the other one is increased or not depending on the distance between two next pixels and the real circle boundary.

Generally, circles generated by the mentioned algorithms present concave and convex boundaries that cannot be contained by the 3×3 prime factors, which we have introduced in the previous section. In Fig. 4 we can see the best discrete approximation of a disk that cannot be decomposed in 3×3 basic structuring elements. It presents some boundaries (marked with ?) that cannot be represented using concave boundaries of type U, J, L, V and r. There is no general method that allows decomposition of such a discrete circle.

Consequently, we have to define the boundary of the approximated disk in such a way that it could be decomposed.

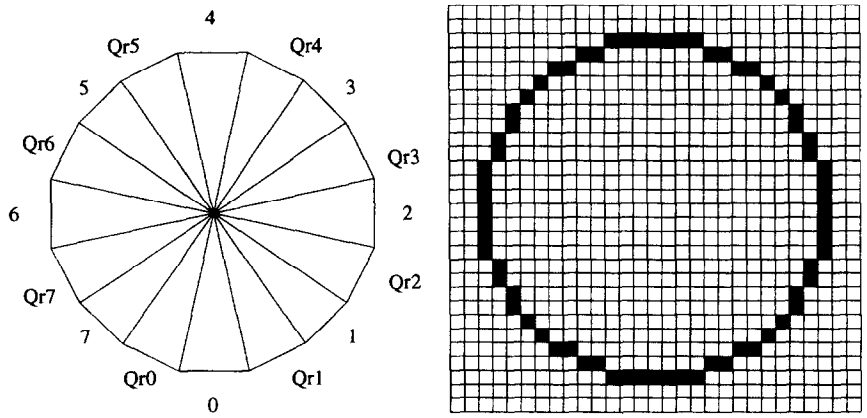
If we do not consider concave boundaries then we can guarantee decomposability (Proposition 2.1). In this case we could approximate a disk by a polygon of eight sides, each side in one of the basic directions of a chain code. But this is not a very good approximation when the radius increases.

To improve the approach we are going to introduce concave boundaries that allow assured 3×3 decomposition. This implies defining a disk approximation whose chain code should be expressed as in Eq. (6). Hence, we only can use the boundaries of Fig. 2(b). Considering the shape of a disk we can state that types U, J, L and V can introduce important errors in the shape of a discrete disk. Only type r boundaries (see Fig. 2(b)) can be useful for discrete approximations of disks.

Therefore, a concave structuring element containing, exclusively, concave boundaries of type r can be expressed as

$$Q_{r0}^{x_0}0^{x_1}Q_{r1}^{x_2}1^{x_3}Q_{r2}^{x_4}2^{x_5}Q_{r3}^{x_6}3^{x_7}Q_{r4}^{x_8}4^{x_9}Q_{r5}^{x_{10}}5^{x_{11}}Q_{r6}^{x_{12}}6^{x_{13}}Q_{r7}^{x_{14}}7^{x_{15}} \tag{7}$$

which can be considered as the boundary of a polygon of sixteen sides, whenever $x_i \neq 0 \forall i$. Eight sides are given by the basic eight directions of a chain code. And the other eight $\{Q_{ri}^{x_i}\}, \forall i \in \{0,\ldots,7\}$ are approximations of



(a) (b)

Fig. 5. (a) Example of a hexadecagon denoting the direction and boundary type of each side; (b) discrete hexadecagon approximating a disk of radius 12. The parameters of its chain code are $a = 2$, $b = 6$ and $c = 3$.

segments with slopes $-1/2$, $1/2$, 2, -2, $-1/2$, $1/2$, 2, -2, respectively.

The above discussion leads us to state that only a hexadecagon can approximate a disk that can be decomposed in 3×3 structuring elements. To ameliorate the discrete approximation of the disk implies decomposition in prime factors larger than 3×3.

To maintain the isotropy of the circle we have imposed the symmetry of each side of the polygon with respect to the center of the element; that is,

$$\begin{aligned} a &= x_0 = x_2 = x_4 = x_6 = x_8 = x_{10} = x_{12} = x_{14} \\ b &= x_1 = x_5 = x_9 = x_{13} \\ c &= x_3 = x_7 = x_{11} = x_{15} \end{aligned}$$

Then the boundaries of the defined hexadecagon are expressed as

$$Q^a_{r0}0^b Q^a_{r1}1^c Q^a_{r2}2^b Q^a_{r3}3^c Q^a_{r4}4^b Q^a_{r5}5^c Q^a_{r6}6^b Q^a_{r7}7^c$$

In Fig. 5(a) we show a polygon and for each side the corresponding direction expressed by the chain code of Eq. (7). Moreover, in Fig. 5(b) we show an approximation of a disk with a radius of twelve pixels.

As has been defined, for a given radius there can be some combinations of values for a, b and c, that define polygons of 16 sides. Therefore, in order to get optimal decomposable disks it is interesting to find the combination that minimizes the approximation error between the polygon and the circle.

3.1. Fitting polygons by a least-squares approach

In this section we develop a fast algorithm to find the parameters that define an optimal disk. It is the result of minimizing the error between the hexadecagon and a circle of the same radius. Minimization will be made by a least-squares approach.

Firstly, we define the error function between the boundary of a disk of radius R and a polygon with a, b and c parameters. Provided that the disk presents symmetry with respect to the center, we will only consider the error function for a quarter of a disk:

$$E(a,b,c) = \|\boldsymbol{F} - \hat{\boldsymbol{F}}\|^2 = \sum_{x=-\left(\frac{b}{2}+2a\right)}^{R-a-1} [F(x) - \hat{F}(x)]^2$$

where $F(x) = \sqrt{R^2 - x^2}$, and $\hat{F}(x)$ is the value for the y coordinate on the polygon boundary that we want to construct. It must be pointed out that c can be expressed as $c = R - [(b/2) + 3a]$, so it does not appear in the previous expression.

The expression can be divided into three parts:

$$\begin{aligned} E(a,b,c) &= 2E_B(b') + 2E_A(a,b') + E_C(a,b') \\ &= 2\sum_{x=1}^{b'} F(x)^2 + 2\sum_{x=b'+1}^{b'+2a} [F(x) - k(x,b')]^2 \\ &\quad + \sum_{x=b'+2a+1}^{R-a-1} [F(x) - (x - b' - a)]^2 \end{aligned}$$

where $b' = b/2$, and

$$k(x,b') = \begin{cases} \dfrac{x-b'}{2} & \text{if } x-b' \text{ is even} \\ \dfrac{x-b'+1}{2} & \text{if } x-b' \text{ is odd} \end{cases}$$

The process to obtain the parameters a, b and c, that minimize this function can be considered as a constraint satisfaction algorithm with the following constraints:

$$\begin{aligned} b' &\in \{0, \ldots, R\} \\ a &\in \{0, \ldots, (R-b')/3\} \end{aligned}$$

The algorithm can be constructed exploiting the recursive expression of partial errors; that is

$$E_B(b') = \sum_{x=1}^{b'} F(x)^2 = E_B(b'-1) + F(b')^2 \qquad (8)$$

$$
\begin{aligned}
E_A(a,b') &= \sum_{x=b'+1}^{b'+2a} [F(x)-k(x,b')]^2 \\
&= E_A(a-1,b') + [F(b'+2a)-k(b'+2a,b')]^2 \\
&\quad + [F(b'+2a-1)-k(b'+2a-1,b')]^2 \\
&= E_A(a-1,b') + [F(b'+2a)-a]^2 \\
&\quad + [F(b'+2a-1)-a]^2
\end{aligned} \tag{9}
$$

thus, we can formulate the following algorithm.

Algorithm 3.1. The values of a, b and c that minimize function E can be obtained as follows:

Step 1. Construct a table storing values for $F(x)\forall x \in \{0,\ldots, R\}$, where $F(x)=\sqrt{R^2-x^2}$.

Construct a table storing values for $E_B(b')\forall b' \in \{0,\ldots, R\}$, computed recursively using Eq. (8).

Construct a table storing values for $E_A(a,b')\forall(a,b') \in \{0,\ldots,(R-b')/3\} \times \{0,\ldots,R\}$, computed recursively using Eq. (9).

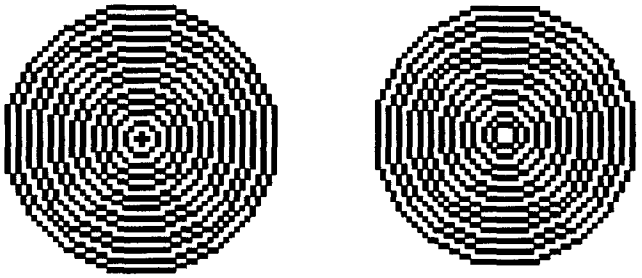
Step 2. For each $(a,b') \in \{0,\ldots,(R-b')/3\} \times \{0,\ldots, R\}$, compute

$$
E(a,b,c) = 2E_B(b') + 2E_A(a,b') + \sum_{x=b'+2a+1}^{R-a-1} [F(x)-(x-b'-a)]^2
$$

using the constructed tables in previous steps. A pair (a,b') is stored if it minimizes E for all (a,b').

Step 3. Given the pair (a,b'), that is the result of the previous step, return a, $b=2b'$ and $c=R-3a-b$ as the solution.

Some results are given in Table 1. The polygons represented by these results are shown in Fig. 6. In this table, we give the total error $E(a,b,c)$ for a quarter of a polygon and the average error $\bar{E}$, that is, error per point. In the same table we provide the error per point for an optimal discrete approximation of a circle. This approximation has been constructed by a generator algorithm minimizing the error using a midpoint criterion.



(a) (b)

Fig. 6. Hexadecagons fitting a decomposable disk with minimum square error: (a) disks with odd radius value $R = 1,\ldots, 25$; (b) disks with even radius value $R = 2,\ldots,24$.

From these we can see that for the first 25 radii the polygonal approximation generally provides the same result as an optimal discrete approximation of circles. Evidently, when the radius increases, the error increases, since to approximate a circle by a hexadecagon supposes some errors, as we can see in Fig. 7(b) and Fig. 7(c).

To evaluate the disk approximation we can compare it with the isotropic morphological operations that can be computed using distance transformations on binary images. Considering chamfer metrics we can state that the obtained disk by our approach has an error that is between the errors given by 5×5 chamfer metrics and 7×7 chamfer metrics given in [13].

Although some errors are introduced, decomposability by 3×3 elements is quite an important property. Now, we are going to prove that these approximated disks are always decomposable.

3.2. Decomposition of disks

By Proposition 2.2, we have to solve a linear equation system in order to find a decomposition of a concave structuring element. Then, for a given disk S, defined by

$$
S = Q^a_{r0}0^b Q^a_{r1}1^c Q^a_{r2}2^b Q^a_{r3}3^c Q^a_{r4}4^b Q^a_{r5}5^c Q^a_{r6}6^b Q^a_{r7}7^c
$$

we have to construct the corresponding linear system. Afterwards, we have to find the solution that assures

Table 1
Results of the developed algorithm for some given radii. For each radius we give the parameters a, b and c defining the optimal decomposable disk. Furthermore, we give the error function $E(a,b,c)$. In order that these results could be compared to a discrete optimal circle, we give the average error for each point in the decomposable hexadecagons, $\bar{E}_{\text{Hex}}$, and the average error for the circles generated by a midpoint generator algorithm, $\bar{E}_{\text{Mid}}$

Radius	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
a	0	0	0	0	0	0	1	1	2	1	1	2	2	2	2	3	3	3	3	3	4	4	3	4	4
b	0	2	2	2	4	4	4	4	4	6	6	6	6	8	8	8	8	8	10	10	10	10	12	12	12
c	1	1	2	3	3	4	2	3	1	4	5	3	4	4	5	3	4	5	5	6	4	5	8	6	7
$E(a,b,c)$	0	0.14	0.11	0.66	0.36	0.61	0.53	0.63	1.22	0.81	1.31	1.00	1.44	1.97	1.38	2.13	1.74	2.67	2.67	2.22	3.05	2.87	4.06	3.59	3.45
$\bar{E}_{\text{Hex}}$	0	0.07	0.03	0.11	0.06	0.08	0.05	0.06	0.10	0.06	0.09	0.06	0.08	0.10	0.06	0.09	0.07	0.11	0.10	0.08	0.10	0.09	0.13	0.10	0.10
$\bar{E}_{\text{Mid}}$	0	0.07	0.03	0.11	0.06	0.08	0.05	0.06	0.10	0.06	0.09	0.06	0.07	0.08	0.05	0.09	0.07	0.08	0.07	0.06	0.08	0.06	0.09	0.06	0.07

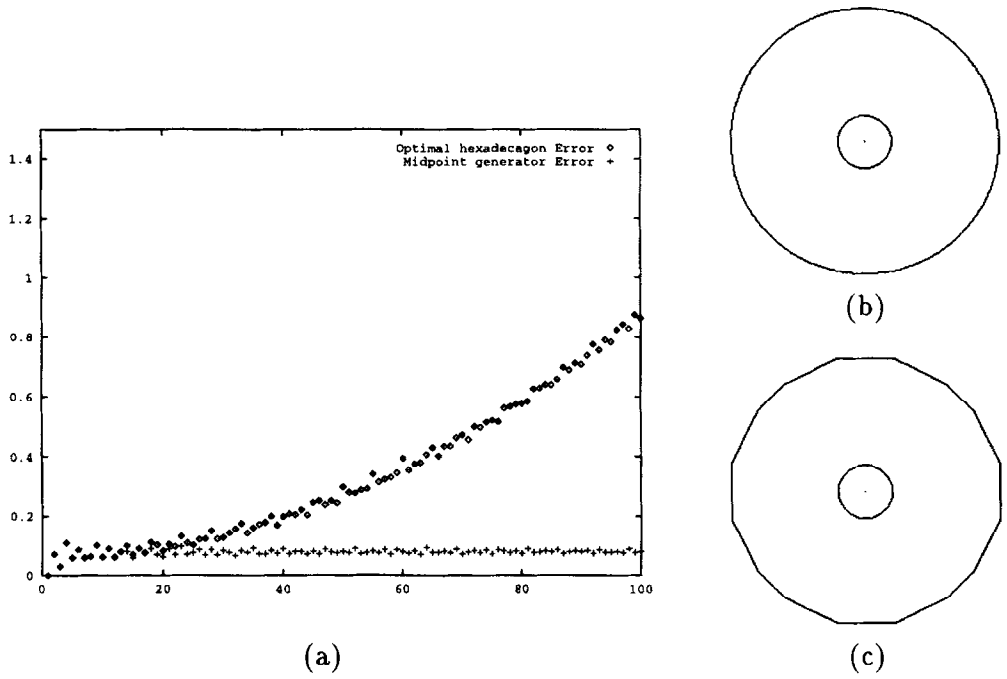


Fig. 7. (a) Graphical comparison between the error of the obtained decomposable hexadecagons and the best discrete approximation of circles with a midpoint generator algorithm; (b) example of two optimal disks generated by a midpoint algorithm with radii 20 and 100; (c) example of two optimal decomposable hexadecagons with radii 20 and 100.

decomposability for any structuring element having this shape.

Firstly, we have to take from the set $\{A_i\}$, introduced in Section 2, all possible concave prime factors for S, that is, we select every structuring element that has all its boundaries contained in S. These structuring elements are shown in Fig. 8. We have 56 prime factors with boundaries that can be contained by a hexadecagon. They are grouped in families of the same shape.

Therefore, these elements are all the possible prime factors we can use to prove decomposability for the defined disks. We will denote them as $\{C_i\}\forall i \in \{1,\ldots,56\}$, and from them we can construct the matrices $\boldsymbol{\Theta}$ and $\boldsymbol{\Omega}$, with dimensions 56×8, as specified in Proposition 2.2.

Secondly we have to construct the vectors $\boldsymbol{Y}$ and $\boldsymbol{Z}$ from S, which are

$$\boldsymbol{Y} = (a,a,a,a,a,a,a,a)$$

$$\boldsymbol{Z} = (b,c,b,c,b,c,b,c)$$

independently of the radius. Then, the linear system to be solved is

$$\begin{aligned} \boldsymbol{\Theta X} &= \boldsymbol{Y} \\ \boldsymbol{Z} - \boldsymbol{B} &= \boldsymbol{\Omega X} \\ b_7 + b_0 + b_1 &= b_3 + b_4 + b_5 \\ b_1 + b_2 + b_3 &= b_5 + b_6 + b_7 \end{aligned} \tag{10}$$

where $\boldsymbol{X} = (x_1,\ldots,x_{56})$ is the vector of variables whose solution values represent the number of dilations for each structuring element being a concave prime factor of S.

Also, $\boldsymbol{B} = (b_0,b_1,b_2,b_3,b_4,b_5,b_6,b_7)$ is a vector of variables too, whose solution represents the chain code of the convex structuring element B. Two last equations in Eq. (10) force the solutions for b_i to represent a valid chain code.

Hence, the solution of this system has to provide with the following decomposition:

$$S = x_1 C_1 \oplus \ldots \oplus x_{56} C_{56} \oplus B$$

but the problem is to find the solution, given that Eq. (10) is an undetermined system. It is formed by 64 variables and 18 equations, which imply a large number of solutions.

To avoid this indetermination, we have considered each family separately, constructing nine matrices $\boldsymbol{\Theta}$ and $\boldsymbol{\Omega}$, one for each family of basic structuring elements. Every family assures that every concave boundary of S will be contained in one basic structuring element of the family.

For family 1 the system has been constructed as follows. We take the chain code representation for each basic structuring element of family 1, that is,

$$C_1 = Q_{r0}2^2 Qr5$$

$$C_2 = Qr1Qr46^2$$

$$C_3 = Qr24^2 Qr7$$

$$C_4 = 0^2 Qr3Qr6$$

then from these chain codes we construct the corresponding

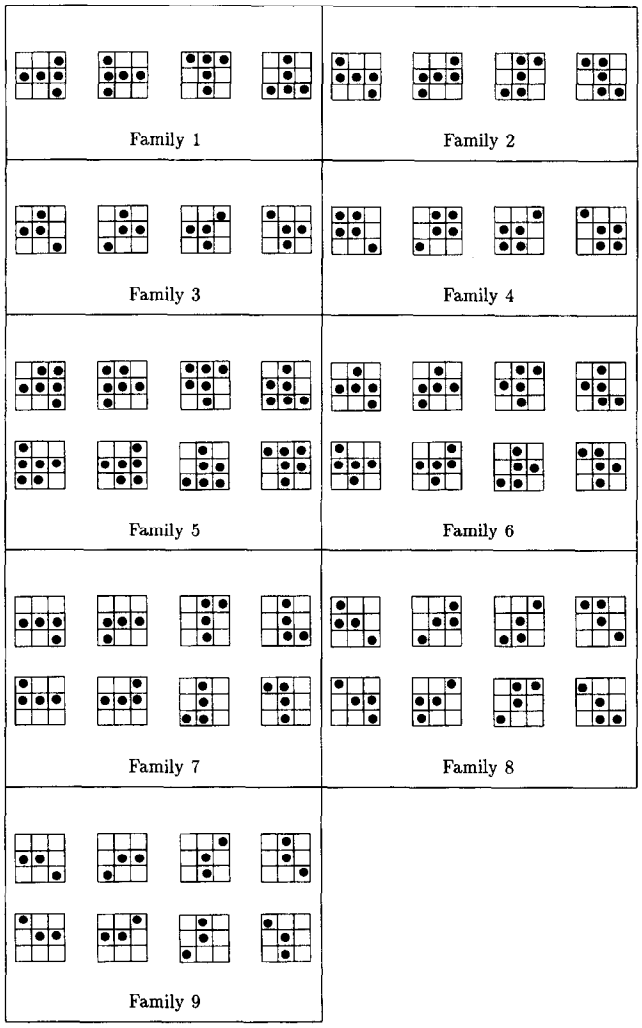


Fig. 8. Set of concave structuring elements with boundaries contained in a discrete approximation of a disk. The structuring elements have been grouped in families, according to the shape.

matrices Θ and Ω, for concave and convex boundaries, respectively,

$$\Theta = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad \Omega = \begin{pmatrix} 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

The solution of Eq. (10) considering the above matrices is

$$X = (a, a, a, a)$$

$$B = (b-2a, c, b-2a, c, b-2a, c, b-2a, c)$$

which implies

$$S = aC_1 \oplus aC_2 \oplus aC_3 \oplus aC_4 \oplus (b-2a)Q1 \oplus cQ_2 \oplus (b-2a)Q_3 \oplus cQ_4$$

The same process over all families implies to solve a linear system with 18 equations and 12 variables for families 1 to 4, and 18 equations with 16 variables for families 5 to 9. In every case the system is overdetermined. For every linear system we have obtained one unique solution that is given in Table 2. The solutions are given in terms of the shape of the disk, that is, in terms of a, b and c parameters.

4. Results

In this section we will give the algorithm to compute a morphological operation with the obtained results. Afterwards, we analyze the different solutions in terms of computation cost.

Considering the obtained results, to compute a morphological operation implies acting in two steps:

Step 1. For a disk, S, of a given radius apply Algorithm 3.1 obtaining parameters a, b and c.

Step 2. Compute the morphological operation with S using the recursive property of Eq. (2), that is

$$S = aC_1 \oplus \ldots \oplus aC_n \oplus B \tag{11}$$

where $n = 4$ for families 1 to 4, and $n = 8$ for families 5 to 9, the shape of C_i depends on the selected family (see Fig. 8), and

$$B = b_0 Q1 \oplus b_1 Q2 \oplus b_0 Q3 \oplus b_1 Q4 \tag{12}$$

where Qi are given in Fig. 9, this expression is the result of applying the algorithm to decompose convex structuring elements [6].

In the last step there are two parts. The first corresponds to the decomposition with concave structuring elements; in this case the solution is always the same independently of the selected family. The second part is related to the decomposition of a convex structuring element that depends on the values of the $\boldsymbol{B}$ solution, which satisfies an interesting property that simplifies decomposition.

Every $\boldsymbol{B}$ solution verifies $b_0 = b_4$, $b_1 = b_5$, $b_2 = b_6$ and $b_3 = b_7$. Therefore, according to algorithm 1 in [6], decomposition of B only depends on the values of b_0 and b_1, giving Eq. (12). These values are always a function of the a, b and c parameters.

For instance, a disk of radius 16 can be approximated by a hexadecagon with parameters $a = 3$, $b = 8$ and $c = 3$ (see Table 1). Then the decomposition using family 1 is

$$S_{16} = 3A_1 \oplus 3A_2 \oplus 3A_3 \oplus 3A_4 \oplus 2Q1 \oplus 3Q2 \oplus 2Q3 \oplus 3Q4$$

where $b - 2a = 2$ and $c = 3$.

Table 2
Solutions for the decomposition of a disk using different families of concave structuring elements. Left column represents the family; x_i represents the number of dilations with concave element C_i of the corresponding family; b_i gives the chain code of the resulting convex structuring element

Family	$X = (x_1,...,x_n)$	$B = (b_1,...,b_N)$
1	$X = (a,a,a,a)$	$B = (b-2a,c,b-2a,c,b-2a,c,b-2a,c)$
2	$X = (a,a,a,a)$	$B = (b-2a,c,b-2a,c,b-2a,c,b-2a,c)$
3	$X = (a,a,a,a)$	$B = (b,c-a,b,c-a,b,c-a,b,c-a)$
4	$X = (a,a,a,a)$	$B = (b-2a,c,b-2a,c,b-2a,c,b-2a,c)$
5	$X = (a,a,a,a,a,a,a,a)$	$B = (b-5a,c-2a,b-5a,c-2a,b\text{-}5a,c-2a,b-5a,c-2a)$
6	$X = (a,a,a,a,a,a,a,a)$	$B = (b-2a,c-4a,b-2a,c-4a,b-2a,c-4a,b-2a,c-4a)$
7	$X = (a,a,a,a,a,a,a,a)$	$B = (b-6a,c,b-6a,c,b-6a,c,b-6a,c)$
8	$X = (a,a,a,a,a,a,a,a)$	$B = (b-2a,c-4a,b-2a,c-4a,b-2a,c-4a,b-2a,c-4a)$
9	$X = (a,a,a,a,a,a,a,a)$	$B = (b-2a,c-2a,b-2a,c-2a,b-2a,c-2a,b-2a,c-2a)$

As has been analyzed in previous works [6], this decomposition of B can introduce some shifts. These can be avoided by changing the center of the basic structuring elements and introducing some additional shifts.

Considering the obtained solutions, we can note that every one verifies $b_0 = b_2 = b_4 = b_6$ and $b_1 = b_3 = b_5 = b_7$. Hence, we can avoid unnecessary shifts by using the elements Qi given in Fig. 9 and replacing the following dilations in Eq. (12):

$$b_0Q1 \oplus b_0Q3 = kQ1 \oplus kQ1' \oplus kQ3 \oplus kQ3'$$

if $b_0 = 2k$, and

$$b_0Q1 \oplus b_0Q3 = (k+1)Q1 \oplus kQ1' \oplus (k+1)Q3 \oplus kQ3' \oplus \{(1,0)\}$$

if $b_0 = 2k+1$, and

$$b_1Q2 \oplus b_1Q4 = pQ2 \oplus pQ2' \oplus pQ4 \oplus pQ4'$$

if $b_1 = 2p$, and finally

$$b_1Q2 \oplus b_1Q4 = (p+1)Q2 \oplus pQ2' \oplus (p+1)Q4 \oplus pQ4' \oplus \{(0,1)\}$$

if $b_1 = 2p+1$; $\{(x,y)\}$ denotes the shift to be applied.

Thus far, we have exposed how to decompose a disk given by a, b and c parameters, using a specific family of basic structuring elements. Now, we are going to analyze how each family reaches the decomposition.

By definition, every solution must satisfy $x_i \geq 0$ and $b_i \geq 0$, for all i. Hence, firstly we will consider which constraints introduce the previous conditions to enable decomposition by each family. As we can see in Table 2, the condition on x_i is fulfilled for every family. Otherwise, for the second condition we have to add some constraints.

Secondly, we will take into account the computation complexity of each solution. We give the total number of points in all the structuring elements of the decomposition as a computation cost evaluator. This number will be denoted by λ.

Table 3 gives the set of constraints added and the cost evaluator, λ, for each family. As we can note in this table, better costs are given by families 7, 8 and 9. However, at the same time these families imply some very restrictive constraints.

Constraints $b \geq 5a$, $c \geq 4a$, $c \geq 2a$ are not satisfied for the disk approximation given by the developed algorithm (Table 2). Then families 3, 5, 6, 7, 8 and 9 cannot assure decomposability.

Only families 1, 2 and 4 impose constraints that are satisfied by the approximated disks. In effect, we have proved that these constraints are satisfied by all disks from radius 1 to 500. Although we have not proved global decomposability, we think it is a good result considering that in image processing applications greater radii are not frequently used.

In summary, we can take families 1, 2 and 4 separately, as sets of prime factors that assure decomposability of optimal hexadecagons approximating disks, and with a computational cost of $4(3a + b + c)$. In general, the decomposition

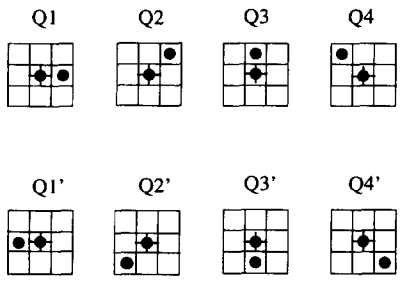


Fig. 9. Set of basic prime factors to decompose convex structuring elements.

Table 3
Constraints and computation costs imposed by each family

Family	Constraints		λ
1,2,4	$b \geq 2a$	$c \geq 0$	$4(3a+b+c)$
3	$b \geq 0$	$c \geq a$	$4(3a+b+c)$
5	$b \geq 5a$	$c \geq 2a$	$4(5a+b+c)$
6	$b \geq 2a$	$c \geq 4a$	$4(4a+b+c)$
7	$b \geq 6a$	$c \geq 0$	$4(2a+b+c)$
8	$b \geq 2a$	$c \geq 4a$	$4(2a+b+c)$
9	$b \geq 2a$	$c \geq 2a$	$4(2a+b+c)$

for a given S hexadecagon of parameters a, b and c is

$$S = aC_1 \oplus aC_2 \oplus aC_3 \oplus aC_4 \oplus (b-2a)Q1 \oplus cQ2 \oplus (b-2a)Q3 \oplus cQ4$$

where C_i represents the elements of one family, 1, 2 or 4 (see Fig. 8), and Qi are shown in Fig. 9.

5. Conclusions

In this work we present an algorithm to obtain 3×3 decompositions of disks. Isotropic structuring elements present interesting properties in computing morphological operations. For this reason we have developed a method to decompose disks of any given radius to improve the computation of these morphological transformations.

First we have developed a constraint-satisfaction algorithm to obtain the decomposable element which better approximates a circle of arbitrary radius by a least-squares approach. The algorithm exploits the recursive error expression, storing computed partial path costs.

A disk is approximated by a hexadecagon which is the best disk approximation that allows 3×3 decomposition. The shape of the disk is expressed by three parameters a, b and c. These parameters represent the sides of the approximated disk. The values b, $2a$ and c indicate the number of pixels of sides with slopes 0, 1/2, 1, respectively. Considering a specific symmetry we can construct the complete polygon from the given three sides.

These parameters are also used to find the decomposition of a disk. For this purpose, we have presented some families of basic structuring elements that can decompose hexadecagons. We have analyzed the computation cost of the decomposition result for each family, as well as the constraints imposed in order to assure decomposition.

Consequently, we have selected three families, 1, 2 and 4 shown in Fig. 8, that assure the decomposition of approximated disks for a wide range of radii. Computation cost, in terms of number of pixels of the complete decomposition, is $4(3a + b + c)$.

Acknowledgements

This work has been partially funded by the project CICYT TAP96-0629-C04-03.

References

[1] J. Serra, Image Analysis and Mathematical Morphology, Academic Press, London, 1982.
[2] J. Serra, Image Analysis and Mathematical Morphology II: Theoretical Advances, Academic Press, London, 1988.
[3] L. Abbott, R.M. Haralick, Pipeline architectures for morphologic image analysis, Machine Vision and Applications 1 (1988) 23–40.
[4] X. Zhuang, R.M. Haralick, Morphological structuring element decomposition, Computer Vision, Graphics and Image Processing 35 (1986) 370–382.
[5] X. Zhuang, Decomposition of morphological structuring elements, Journal of Mathematical Imaging and Vision 4 (1) (1994) 5–18.
[6] J. Xu, Decomposition of convex polygonal morphological structuring elements into neighborhood subsets, IEEE Trans. Patt. Anal. Machine Intell. 13 (2) (1991) 153–162.
[7] H. Park, R.T. Chin, Decomposition of arbitrarily shaped morphological structuring elements, IEEE Trans. Patt. Anal. Machine Intell. 17 (1) (1995) 2–15.
[8] P. Maragos, Pattern spectrum and multiscale shape representation, IEEE Trans. Patt. Anal. Machine Intell. 11 (7) (1989) 701–715.
[9] M. Vanrell, X. Roca, J. Vitrià, A general morphological framework for perceptual texture discrimination based on granulometries. In: J. Serra, P. Salembier (Eds.), Mathematical Morphology and its Applications to Signal Processing, UPC Publications Office, Barcelona, 1993, pp. 112–118.
[10] P. Maragos, M. Akmal, Partial differential equations in image analysis: continuous modeling, discrete processing. In: P. Delogne (Ed.), International Conference on Image Processing, IEEE, Laussane, 1996, pp. 61–64.
[11] M. Vanrell, X. Roca, J. Vitrià, A multidimensional scaling approach to explore the behavior of a texture perception algorithm, Machine Vision and Applications, 1997 (9) 262–271.
[12] M. Vanrell, Exploring the behavior of a texture perception algorithm, Technical Report 12 (CVC, Computer Vision Center, Barcelona, 1996).
[13] G. Borgefors, Distance transformations in digital images, Computer Vision, Graphics and Image Processing 34 (1986) 344–371.
[14] H. Park, R.T. Chin, Optimal decomposition of convex morphological structuring elements for 4-connected parallel array processors, IEEE Trans. Patt. Anal. Machine Intell. 16 (3) (1994) 304–313.
[15] J.D. Foley, A. van Daro, S.K. Feiner, J.F. Hughes, R.L. Phillips, Computer Graphics, Principles and Practice, Addison-Wesley, 1990.