A fast algorithm for sequential machines to compute pattern spectrum via chess-board distance transform

P. Ghosh [a,*], B. Chanda [b]

[a] *Department of Mathematics, Jadavpur University, Calcutta 700 032, India*
[b] *National Centre for Knowledge Based Computing, Electronics and Communication Sciences Unit, Indian Statistical Institute, 203 B.T. Road, Calcutta 700 035, India*

Received 28 October 1993; revised 27 June 1994

Abstract

Computing pattern spectrum is one of the key methods for determining shape-size distribution of objects in the image. Morphological algorithm for computing the pattern spectrum by successive opening and area computing is very costly on sequential machines. In this paper a fast algorithm for the same using the chess-board distance transform is proposed.

Keywords: Mathematical morphology; Binary objects; Distance transform; Pattern spectrum

1. Introduction

The problem of shape representation and shape-size description is very important in computer vision and image analysis. An algebraic system of operators, such as those of mathematical morphology, are useful for the purpose because when acting on complex shapes they are able to decompose them into their meaningful parts and separate them from extraneous parts as in (Giardina and Dougherty, 1990; Matheron, 1975; Maragos, 1989; Haralick and Shapiro, 1992; Serra, 1982). Serra (1982) has extensively used these size distributions in image analysis applications to petrography and biology. Maragos (1989) has viewed the size distribution via a concept of pattern spectrum and has extended the pattern spectrum to gray-tone images and arbitrary multi-level signals. Morphological operators are suitable for parallel implementation and morphological algorithms are very fast on parallel machines because simple and identical operations are applied simultaneously to many points. Due to the same reason they, including the algorithm for computing the pattern spectrum, are usually very slow on sequential machines. Here we propose a fast algorithm suitable for sequential machines for computing the pattern spectrum with respect to a family of circles using the chess-board distance transform matrix (as in (Ghosh and Chanda, 1993)).

This paper is organized as follows. In Section 2 we present some basic definitions of morphological operations and describe the pattern spectrum as in (Serra, 1982; Maragos, 1989). The proposed method is described in Section 3. Section 4 presents the implementation and computational complexity. Section 5 presents the concluding remarks.

2. Preliminaries

Monochrome image or simply image, refers to a two-dimensional intensity function $f(x, y)$, where x and y denote spatial coordinates and the value of $f(x, y)$ at any point (pixel) is proportional to the brightness (or gray-value) of the image at the point. Here we deal with binary images only. More precisely we deal with 2-D objects represented as a maximal set of connected points having gray-value 1.

Let A, B be subsets of $\mathbb{R}^2$ or $\mathbb{Z}^2$, where $\mathbb{R}$ and $\mathbb{Z}$ are the set of real and integer numbers, respectively.

(i) *Dilation* $$D(A, B) = \{a+b \mid a\in A,\ b\in B\} = \bigcup_{b\in B} \{A+b\} \tag{1}$$

where $A+p = \{a+p \mid a\in A\}$ denotes the translate of A by the vector p.

(ii) *Erosion* $$E(A, B) = \{a \mid B+a \subseteq A\} = \bigcap_{b\in B} \{A-b\} \tag{2}$$

(iii) *Opening* $$O(A, B) = D(E(A, B), B) \tag{3}$$

(iv) *Closing* $$C(A, B) = E(D(A, B), B) \tag{4}$$

B is called the structuring element.

Let B be a compact subset of the plane $\mathbb{R}^2$; such B is called a continuous space binary pattern. Let B have size one. Then the set

$$rB = \{rb \mid b\in B\} \tag{5}$$

defines a homotopic pattern of size r, where r is any nonnegative real number. The shape of rB is the same as that of B. Now, if B is a discrete space binary pattern, i.e. a finite connected subset of the discrete plane $\mathbb{Z}^2$, then

$$nB = \underbrace{D(D(\cdots D}_{n\text{ times}}(D(D(B, B), B), B), \ldots), B), B)\ . \tag{6}$$

Eq. (6) defines a family of binary patterns generated by B parameterized by the discrete size parameter n ($n=0, 1, 2, \ldots$). By convention if $n=0$, $nB = \{(0, 0)\}$. Multi-scale opening of A by B at scale $n=0, 1, 2, \ldots$ is the opening of A by nB:

$$O(A, nB) = D(E(A, nB), nB)\ . \tag{7}$$

From (7), (1) and (2) it follows that

$$O(A, nB) = \bigcup_{(nB+p)\subseteq A} (nB+p)\ . \tag{8}$$

Hence $O(A, nB)$ eliminates from A all objects or their parts inside which nB cannot fit. That is why the size n of nB is referred to as synonymous with the scale at which the filter $O(A, nB)$ operates. The pattern spectrum provides a measure of similarity between a set and the collection of all structuring elements nB. The pattern spectrum $PS_A(n, B)$ of a set A in terms of the structuring element B is given by Maragos (1989) as

$$PS_A(n, B) = \#[O(A, nB) - O(A, (n+1)B)] \quad \text{for } n\geqslant 0 \tag{9}$$

where $n\in\mathbb{Z}$ and $\#[\cdot]$ denotes the cardinality of any set. Since opening is an anti-extensive operation and nB is increasing as n increases,

$$O[A, (n+1)B] \subseteq O[A, nB] \quad \text{for } n \geqslant 0 . \tag{10}$$

Thus $\#[O(A, nB)]$ is nonnegative and is decreasing as n increases. Since A is finite, there exists a positive integer $N=\max\{n \mid O(A, nB) \neq \emptyset\}$. Then $\#[O(A, nB)]=0$ for all $n>N$.

Eq. (9) suggests that the pattern spectrum can be computed by successive opening followed by set subtraction followed by area computing (pixel counting) procedure. The method of computing the pattern spectrum by direct implementation of Eq. (9) is referred to as "conventional method". It is evident that the conventional method is very slow if implemented on a sequential machine, at least when N is reasonably large. Here we suggest a fast algorithm for computing the pattern spectrum using the distance transform of the binary image. A detailed discussion of the distance transform can be found in Rosenfeld and Kak (1982).

Since 8-connectivity for objects is popular, in this paper we consider the chess-board distance defined as $d(p, q)=\max\{|x-u|, |y-v|\}$ where (x, y) and (u, v) are coordinates of p and q respectively. To each pixel in a given set A, the distance transform assigns a number which is the minimum distance between the pixel and A^c (i.e., the complement of A). For a binary image these numbers form a matrix DT whose elements are: $DT(x, y)=\min_{(u,v)}\{d(p, q)\}$ such that $p(x, y)\in A$ and $q(u, v)\in A^c$. That means if $DT(x, y)=n$ and $B=\{(-1, -1), (-1, 0), (-1, 1), (0, -1), (0, 0), (0, 1), (1, -1), (1, 0), (1, 1)\}$, then at most $(n-1)B$ centered at (x, y) can fit in A. In other words, $O(A, nB)=\bigcup\{nB+(x, y)\}$ for all (x, y) such that $DT(x, y)\geqslant n+1$. Secondly, $\max_{x,y} DT(x, y)=N+1$. Therefore, $\#[O(A, nB)]$ can be computed from the distance transform matrix DT and $\#[nB]$.

3. The proposed method

In this section the mathematical basis of the proposed algorithm is presented. Let us recall Eq. (9) which can be rewritten as:

$$PS_A(n, B)=\#[O(A, nB)]-\#[O(A, (n+1)B)] \quad \text{for } 0 \leqslant n \leqslant N .$$

Foreground A of the binary image may be considered as the union of A_1, A_2, etc. We know, if $A_1, A_2, \ldots, A_m$ are m sets then

$$\begin{aligned}&\#(A_1 \cup A_2 \cup \cdots \cup A_m)\\ &\quad = \sum \#A_k - \sum \#(A_i \cap A_j) + \sum \#(A_i \cap A_j \cap A_k) \cdots + (-1)^m \#(A_1 \cap A_2 \cap \cdots \cap A_m) .\end{aligned} \tag{11}$$

In the present context, the set A_i is defined as follows. Consider any arbitrary distance transform value l, then

$$A_i=\{(x, y) \mid \max\{|x-u|, |y-v|\}<l \,\&\, DT(u, v)\geqslant l\} . \tag{12}$$

That is, A_i $(1 \leqslant i \leqslant m)$ consists of the pixels lying within an upright square of size $(2l-1)^2$ around (u, v). So, the first term of Eq. (11) is $\sum \#A_k = m*(2l-1)^2$.

Now let (x_i, y_i) and (x_j, y_j) be the coordinates of the central pixel of A_i and A_j, respectively. So the number of pixels in the set $(A_i \cap A_j)$ is

$$[2l+1-|x_i-x_j|]*[2l+1-|y_i-y_j|] .$$

For each A_i $(1 \leqslant i \leqslant m)$ we take A_j $(1 \leqslant j \leqslant m)$, so that the number of $(A_i \cap A_j)$ considering all i and j $(i \neq j)$ is $\binom{m}{2}$. Then the second term of Eq. (11), i.e. $\sum \#(A_i \cap A_j)$, equals

$$\sum_i \Bigl(\sum_{j\ (j \neq i)} \{2l+1-|x_i-x_j|\}*\{2l+1-|y_i-y_j|\} \Bigr) .$$

Next for the third term of Eq. (11), i.e. $\sum \#(A_i \cap A_j \cap A_k)$, let (x_k, y_k) be the central pixel of A_k. Firstly, the relative distances of each central pixel from the other two are determined. Then we consider the maximum distance along x- and y-direction to compute the number of pixels in $(A_i \cap A_j \cap A_k)$, which is

$$[2l+1-\max\{|x_i-x_j|, |x_i-x_k|, |x_j-x_k|\}] * [2l+1-\max\{|y_i-y_j|, |y_i-y_k|, |y_j-y_k|\}] .$$

For each A_i $(1 \leqslant i \leqslant m)$ we take A_j $(1 \leqslant j \leqslant m)$ and A_k $(1 \leqslant k \leqslant m)$. So the number of pixels of $(A_i \cap A_j \cap A_k)$ considering all i, j and k $(i \neq j, i \neq k, j \neq k)$ is $\binom{m}{3}$. So,

$$\sum \#(A_i \cap A_j \cap A_k) = \sum_{\substack{i \\ (i \neq j \neq k)}} \sum_j \sum_k [2l+1-\max\{|x_i-x_j|, |x_i-x_k|, |x_j-x_k|\}] * [2l+1-\max\{|y_i-y_j|, |y_i-y_k|, |y_j-y_k|\}] .$$

Now by generalizing the last term of Eq. (11), i.e. $(-1)^m \#(A_1 \cap A_2 \cap \cdots \cap A_m)$ can be found as follows.

Let the coordinate of the central pixel of A_m be (x_m, y_m). So the steps are:

Firstly, find out the relative distances of each central pixel from the others along the x-direction (say), that is, find out

$$|x_1-x_2|, |x_1-x_3|, \ldots, |x_1-x_m|, \ldots, |x_i-x_k|, \ldots, |x_{m-1}-x_m| .$$

Secondly, find out the maximum of them, say $d_{x\text{-max}}$ where

$$d_{x\text{-max}} = \max_{1 \leqslant i \leqslant m,\ 1 \leqslant j \leqslant m\ (i \neq j)} \{|x_i-x_j|\} .$$

Similarly we can find out the maximum relative distances along the y-direction, say $d_{y\text{-max}}$. Finally,

$$\#(A_1 \cap A_2 \cap \cdots \cap A_m) = [(2l+1-d_{x\text{-max}})] * [(2l+1-d_{y\text{-max}})] .$$

Different way of interpretation

In our presentation if $DT(x_i, y_i) \geqslant l$ then A_i may be defined as $\{lB+(x_i, y_i)\}$. So the computation of the first term of Eq. (11) is straightforward and the computation of the second term is more or less manageable. But computing the third and the higher terms is very costly because relative positions of many sets are involved. Our basic aim is to reduce the computational cost. If we implement the above idea then the computational cost will be too high. In the following discussion the above idea of computing the cardinality of the combined sets is interpreted in a slightly different way. Now we are interested in finding out the number of additional pixels due to each set A_j given the number of pixels added already due to the set A_i. Earlier we have used $\#(A_i \cup A_j) = \#A_i + \#A_j - \#(A_i \cap A_j)$. Presently we consider, for the same purpose, the relation $\#(A_i \cup A_j) = \#A_i + \#(A_j - A_i)$. The idea is that during raster scan of the distance transform as we encounter A_j, i.e. $DT(x, y) \geqslant l$, we search for the nearest A_i's in the previous rows and in the same row but previous columns, and once found, the number of additional pixels due to A_j is determined depending on the distances.

Algorithm

```
for $1 \leqslant l \leqslant N+1$ do $C_l = 0$;
for all $(x, y)$ if $DT(x, y-1) \geqslant l$ do
if $DT(x, y-1) \geqslant l$ then
    if $DT(x-1, y) \geqslant l$ then $C_l = C_l + 1$;                                   [1]
    else if $DT(x-u, y+v) \geqslant l$ (for $1 \leqslant u, v \leqslant 2l-2$) then $C_l = C_l + u$;   [2]
        else $C_l = C_l + (2l-1)$;                                                      [3]
else if $DT(x-1, y) \geqslant l$ then $C_l = C_l + (2l-1)$                              [4]
```

```
  else if $DT(x-u_1, y-v_1) \geqslant l$ (for $1 \leqslant u_1, v_1 \leqslant 2l-2$) then
    if $DT(x-u_2, y+v_2) \geqslant l$ (for $1 \leqslant u_2, v_2 \leqslant 2l-2$) then
      $C_l = C_l - (2l-1)^2 + (2l-1-u_1)(2l-1-v_1) - (2l-1-u_1)(2l-1-v_1)$;      [5]
    else $C_l = C_l + (2l-1)^2 - (2l-1-u_1)(2l-1-v_1)$;                            [6]
  else if $DT(x-u_2, y-v_2) \geqslant l$ (for $1 \leqslant u_2, v_2 \leqslant 2l-2$) then
      $C_l = C_l + (2l-1)^2 - (2l-1-u_2)(2l-1-v_2)$;                               [7]
    else $C_l = C_l + (2l-1)^2$;                                                     [8]
enddo;
enddo;
```

[Note: "**if** $DT(u, v) \geqslant l$ (**for** $1 \leqslant u, v \leqslant 2l-2$) **then**..." means loop until $DT(u, v) \geqslant l$ is true, and when the condition is satisfied the statement is executed. Secondly, "v" is varied in the inner loop.]

Hence the pattern spectrum is computed as

$$PS_A(l, B) = C_{l+1} - C_{l+2} \quad \text{for } 0 \leqslant l < N \quad \text{and} \quad PS_A(N, B) = C_{N+1} .$$

4. Computational complexity

It may be of interest to compare the computational complexities of the conventional method, the fast algorithm suggested by Bronskill and Venetsanopoulos (1992) and our algorithm for computing the pattern spectrum. Before going to the detailed discussion let us categorize the foreground or the objects in two types: S_1 and S_2. The object in which the pixels having the same distance transform value are connected is called an S_1 *type object*, otherwise it is called an S_2 *type object*. Examples are shown in Fig. 1. The computational complexity for the pattern spectrum is invariant to the types of object when the conventional or Bronskill's method is considered. In our case, however, the type of the object is important.

First we study the computational complexity for the various algorithms for S_1 type objects. Let A be one such object. Let the number of pixels in A be Q and let the number of pixels having distance transform value l be n_l (for $1 \leqslant l \leqslant N+1$). Then obviously

$$n_1 > n_2 > n_3 > \cdots > n_N > n_{N+1} \quad \text{and} \quad n_1 + n_2 + n_3 + \cdots + n_N + n_{N+1} = Q .$$

The radius of a maximal disc centered at (x, y) that can fit in the object is equal to $(DT(x, y) - 1)$, i.e., $(l-1)$ if $DT(x, y) = l$, and the number of pixels in the disc of radius $(l-1)$ is $(2l-1)^2$.

In the *conventional method* the number of operations to erode with a disc of radius $(l-1)$ is $(2l-2)^2 * Q$ and the number of operations to dilate with the same disc is $(2l-1)^2 * \sum_{k=l}^{N+1} n_k$. So the number of operations to open with a disc of radius $(l-1)$ *is* $(2l-1)^2 * [Q + \sum_{k=l}^{N+1} n_k]$. Finally the total number of binary operations to compute the pattern spectrum is

```
(a)                    (b)                    (c)                    (d)
00000000000000000000   00000000000000000000   00000000000000000000   00000000000000000000
01111111111111111110   01111111111111111110   01111100000000111110   01111100000000111110
01111111111111111110   01222222222222222210   01111100000000111110   01222100000000122210
01111111111111111110   01233333333333333210   01111100000000111110   01232100000000123210
01111111111111111110   01222223344332222210   01111111111111111110   01222111111111122210
01111111111111111110   01111122333322111110   01111111111111111110   01111122222222111110
00000111111111100000   00000112222221100000   00000111111111100000   00000111222211100000
00000011111111000000   00000011111111000000   00000011111111000000   00000011111111000000
00000000011000000000   00000000011000000000   00000000011000000000   00000000011000000000
00000000000000000000   00000000000000000000   00000000000000000000   00000000000000000000
```

Fig. 1. (a) S_1 type object. (b) Its distance transform. (c) S_2 type object. (d) Its distance transform.

$$\sum_{l=1}^{N+1}\left[(2l-1)^2 * \left\{Q+\sum_{k=l}^{N+1} n_k\right\}\right]. \tag{13}$$

Hence the order of computational complexity of the Conventional Algorithm is $O((N+1)^4)$.

In the *modified algorithm* suggested by Bronskill and Venetsanopoulos (1988), the structuring element is a disc of radius one. The number of pixels of this disc is 9. Now the number of operations to erode with a disc of radius 1 is $[9*\sum_{k=l}^{N+1} n_k]$ and the number of operations to dilate with a disc of radius $(l-1)$ is $[9*l*\sum_{k=l+1}^{N+1} n_k]$. Thus the number of operations to open with a disc of radius $(l-1)$ is

$$9*\sum_{k=l}^{N+1} n_k + 9*l*\sum_{k=l+1}^{N+1} n_k = 9*[n_l+(l+1)]*\sum_{k=l+1}^{N+1} n_k .$$

Hence the total number of operations to compute the pattern spectrum is

$$\sum_{l=1}^{N+1}\left[9*[n_l+(l+1)]*\sum_{k=l+1}^{N+1} n_k\right]. \tag{14}$$

Hence the order of computational complexity of the modified algorithm is $O((N+1)^4)$.

We first apply the chess-board distance transform algorithm to the binary image and then apply the proposed algorithm for finding the pattern spectrum. Before finding the complexity for the algorithm it is necessary to discuss the computational complexity for the chess-board distance transform. The sequential algorithm to compute the chess-board distance transform requires 9 binary operations and 1 addition operation per pixel. We have numbered the different steps of the proposed algorithm on the right-hand side. Let j be such a step. Let the probability of action j due to distance transform l be denoted by $P_l[j]$.

Then both experiments and intuition suggest that, for an S_1 type object, $P_l[5]=P_l[6]=P_l[7]=0$ for all l. $P_l[2]$ and $P_l[8]$ are very small. Before going into detail let us assume $P_l[1]$ and $P_l[3]$ are equally likely for the distance transform l. Then the number of binary operations for distance transform l is

$$\begin{aligned}
&\left\{2P_l[1]\sum_{k=l}^{N+1} n_k\right\}+\left\{(2+2N/2)\cdot P_l[2]\cdot[1-P_l[1]]\sum_{k=l}^{N+1} n_k\right\}\\
&+\left\{(2+2N/2)\cdot P_l[3]\cdot[1-P_l[1]-P_l[2]]\sum_{k=l}^{N+1} n_k\right\}\\
&+\left\{(2+2N/2)\cdot P_l[4]\cdot[1-P_l[1]-P_l[2]-P_l[3]]\sum_{k=l}^{N+1} n_k\right\}\\
&+\left\{[2+2N/2+(2N/2)^2+2*2N/2]\cdot P_l[8]\cdot[1-P_l[1]-P_l[2]-P_l[3]-P_l[4]]\sum_{k=l}^{N+1} n_k\right\}\\
&\quad=\{2P_l[1]+(N+2)\cdot P_l[2]\cdot[1-P_l[1]]\\
&\qquad+(N+2)\cdot P_l[3]\cdot[1-P_l[1]-P_l[2]]+(N+2)\cdot P_l[4]\cdot[1-P_l[1]-P_l[2]-P[3]]\\
&\qquad+[(3N+2)+N^2]\cdot P_l[8]\cdot[1-P_l[1]-P_l[2]-P_l[3]]-P_l[4]\}\sum_{k=l}^{N+1} n_k
\end{aligned}$$

$$\begin{aligned}&=\{2P_l[1]+(N+2)\cdot[P_l[2]+P_l[3]+P_l[4]]\cdot[1-P_l[1]]\\&\quad-(N+2)\cdot[P_l[3]\cdot P_l[2]-P_l[4]\cdot P_l[2]]-P_l[4]\cdot P_l[3]]\\&\quad+[(N+1)^2+(N+1)]\cdot P_l[8][1-P_l[1]-P_l[2]-P_l[3]]-P_l[4]\}\sum_{k=l}^{N+1}n_k\,.\end{aligned}\tag{15a}$$

Now again $P_l[2] \ll P_l[3]$ and $P_l[8]$ has the order $O(1/N+1)$. So (15a) can be written as

$$\begin{aligned}&\{2P_l[1]+(N+2)(P_l[3][1-P_l[1]]-P_l[3]P_l[2]-P_l[4]\cdot P_l[2]\\&\quad-P_l[4]\cdot P_l[3]\})+[O((N+1)^2)\cdot O(1/N+1)(1-2P_l[1])\}\sum_{k=l}^{N+1}n_k\,.\end{aligned}$$

Hence the order of computational complexity of the proposed algorithm is

$$\begin{aligned}&[O(N+1)+(N+2)\cdot(O(N+1)\cdot O(N+1)-O(N+1)O(N+1)-O(N+1)\cdot O(N+1)\\&\quad-O(N+1)O(N+1))+O((N+1)^2)]\cdot O(N+1)+O((N+1)^3)\,.\end{aligned}$$

Now the number of addition operations is

$$\begin{aligned}&P_l[1]+P_l[2]\cdot(1-P_l[1])+P_l[3]\cdot(1-P_l[1]-P_l[2])\\&\quad+P_l[4]\cdot(1-P_l[1]-P_l[2]-P_l[3])+\cdots\cong O((N+1)^2)\,.\end{aligned}\tag{15b}$$

Hence the total computational complexity from (15a) and (15b) is $O((N+1)^3)$.

Next we study the computational complexity for the various algorithms for S_2 type objects. For S_2 type objects $P_l[5]$, $P_l[6]$, $P_l[7]$ are not zero. So the number of binary operations for these steps are

$$\begin{aligned}&\{[2+(2N+1)/2+(2N/2)^2]P_l[5][1-P_l[1]-P_l[2]-P_l[3]-P_l[4]]\\&\quad+[2+(2N+1)/2+(2N/2)^2]P_l[6][1-P_l[1]-P_l[2]-P_l[3]-P_l[4]-P_l[5]]\\&\quad+[2+(2N+1)/2+(2N/2)^2+2N/2]P_l[7][1-P_l[1]-P_l[2]-P_l[3]-P_l[4]\\&\quad-P_l[5]-P_l[6]]\}\sum_{k=l}^{N+1}n_k\cong O((N+1)^3)\,.\end{aligned}$$

Now again $P_l[2] \ll P_l[3]$ and $P_l[5]$, $P_l[6]$, $P_l[7]$ are small compared to $P_l[3]$ or $P_l[4]$ and $P_l[8]$ has the order $O(1/N+1)$.

Then a similar approach as above suggests that the computational complexity for S_2 type objects of the proposed algorithm is also of the order of $O((N+1)^3)$.

Hence for both S_1 and S_2 types of objects the computational complexity of the algorithm is $O((N+1)^3)$. From this it can be adjudged that the time complexity of the algorithm will be less than for the conventional and modified algorithms.

The pattern spectrum, as defined in Eq. (9), is computed in two steps. First, the distance transform $DT(x, y)$ is computed using the two-pass algorithms designed for sequential machines as in (Rosenfeld and Kak, 1982). Second, the cardinality of the combined sets corresponding to $DT(x, y) \geqslant l$ is computed using the proposed algorithm. Finally, the pattern spectrum $PS_A(n, B)$ is determined. During implementation of the algorithm, a look-up table is used instead of computing the addendum for different local spatial distributions of the pixels (x_i, y_i). The algorithm is tested on a large number of images containing S_1 or S_2 or both types of objects. Computer programs for the proposed algorithms are written (in C) and are executed on a PC-AT-286 on 15 MHz. Some examples are shown in Figs. 2–4, where (a) shows the input image and (b) shows its pattern spectrum. Time requirements due to the different algorithms are listed in Table 1.

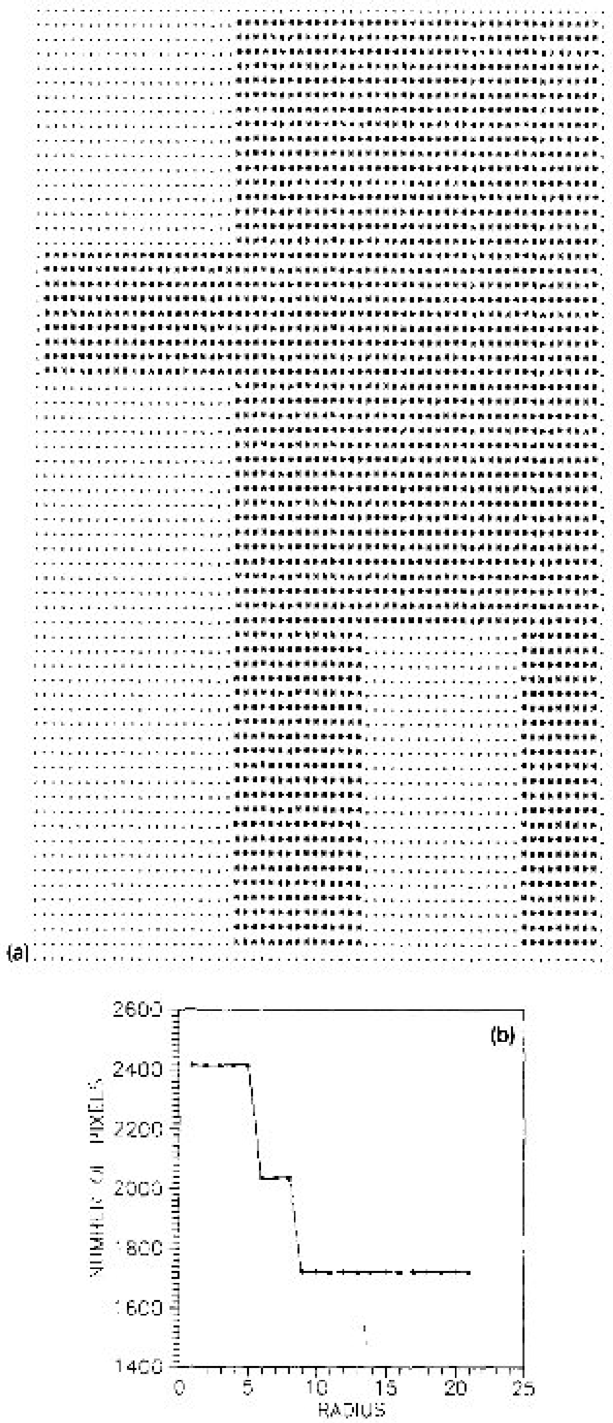


Fig. 2. (a) S_1 type object. (b) Its pattern spectrum.

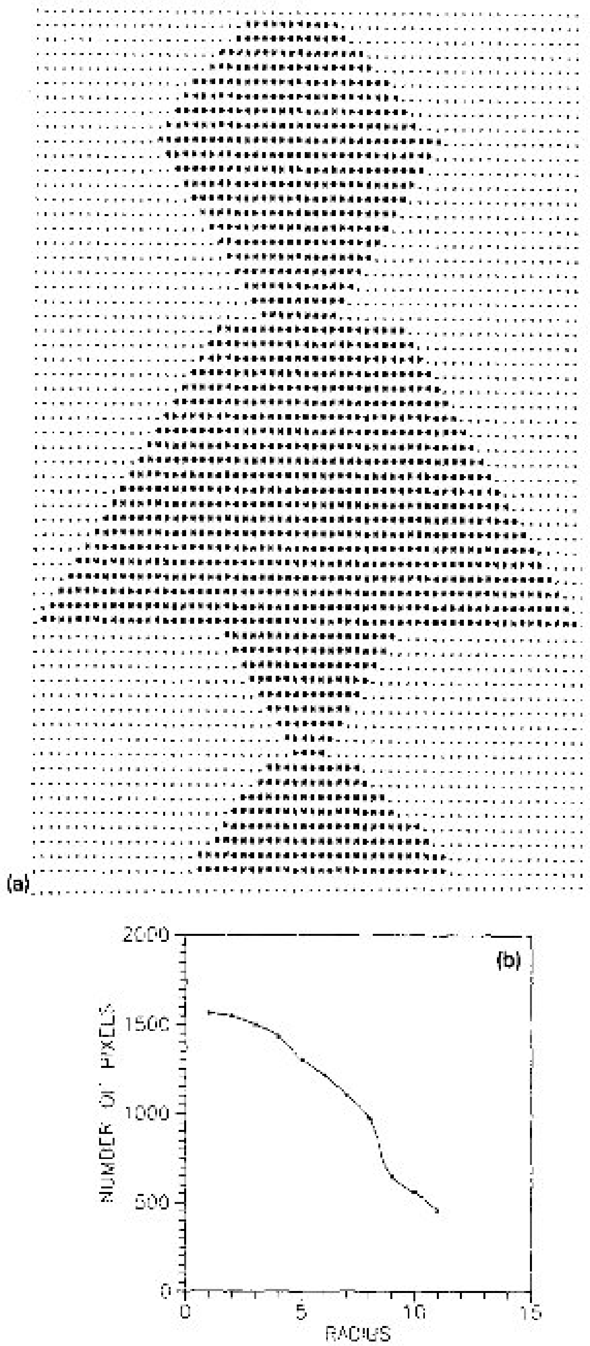
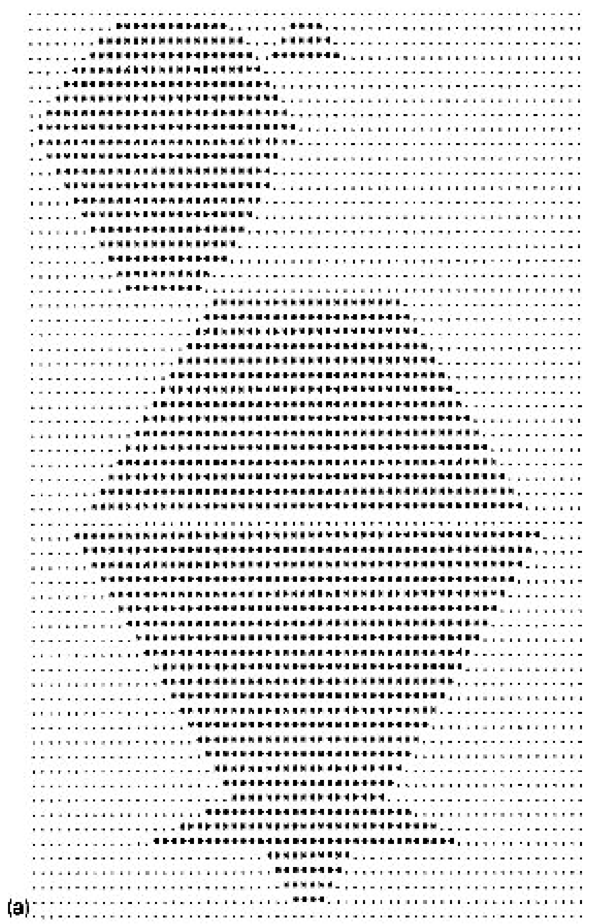


Fig. 3. (a) S_2 type object. (b) Its pattern spectrum.



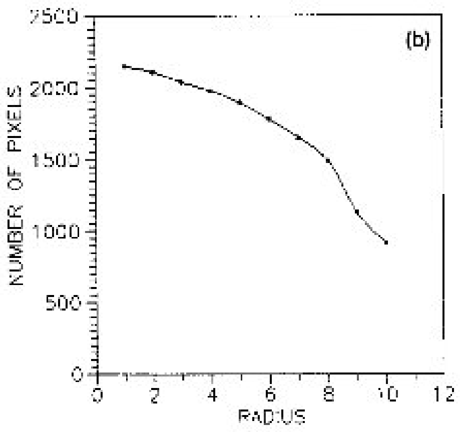


Fig. 4. (a) S_1 and S_2 type object. (b) Its pattern spectrum.

Table 1

Figure no.	Fig. 2	Fig. 3	Fig. 4
Type	S_1	S_2	S_1 & S_2
Size	64×64	64×64	64×64
Proposed	3.90 s	1.54 s	2.21 s
Conventional	20.11 s	14.01 s	16.05 s
Modified (Bronskill)	14.69 s	9.57 s	11.27 s

5. Conclusion

This paper presents a fast algorithm for computing the pattern spectrum for sequential machines. The algorithm is based on the distance transform using the chess-board distance metric, since in most of the image processing works 8-connectivity is assumed for the objects. It is experimentally found that the proposed algorithm (including the sequential method for computing the distance transforms) is much more efficient than the conventional morphological algorithm and the modified algorithm proposed by Bronskill and Venetsanopoulos (1988) for computing the pattern spectrum. Since the structuring element B is convex, it can be decomposed into smaller discs or into simpler structures. In that case the computational efficiency of the conventional method is greatly improved as in (Zhuang and Haralick, 1986; Xu, 1991). However, this approach is not attempted because the order is still not better compared to the proposed method.

Acknowledgement

The authors would like to thank Mr. Amitava Sil, Mr. Supratim Sanyal and Mr. Abhijit Adak for their valuable suggestions and extensive help in the computer implementation.

References

Bronskill, J.F. and A.N. Venetsanopoulos (1988). Multidimensional shape description and recognition using mathematical morphology. *J. Intelligent and Robotic Systems* 1, 117–143.

Ghosh, P. and B. Chanda (1993). Two fast algorithms for sequential machine to compute pattern spectrum with morphological concept. *3rd Internat. Conf. on Advances in Pattern Recognition and Digital Techniques*, Calcutta, India, 28–31 Dec. 1993.

Giardina, C.R. and E.R. Dougherty (1990). *Morphological Methods in Image and Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ.

Haralick, R.M. and L.G. Shapiro (1992). *Digital Image Processing*. Academic Press, New York.

Maragos, P. (1989). Pattern spectrum and multiscale shape representation. *IEEE Trans. Pattern Anal. Machine Intell.* 11 (7), 701–715.

Matheron, G. (1975). *Random Sets and Integral Geometry*. Wiley, New York.

Rosenfeld, A. and A.C. Kak (1982). *Digital Image Processing*, Vol. 2. Academic Press, New York, 2nd edition.

Sconfeld, D. and J. Gautias (1991). Optimal morphological pattern restoration from noisy binary images. *IEEE Trans. Pattern Anal. Machine Intell.* 13 (1), 14–29.

Serra, J. (1982). *Image Analysis and Mathematical Morphology*, Vol. 1. Academic Press, London.

Xu, J. (1991). Decomposition of convex polygonal morphological structuring elements into neighbourhood subsets. *IEEE Trans. Pattern Anal. Machine Intell.* 13 (2), 153–162.

Zhuang, X. and R.M. Haralick (1986). Morphological structuring element decomposition. *Computer Vison, Graphics, and Image Processing* 35, 370–382.