[32] J. H. Conway and J. A. Sloane, "Fast quantizing and decoding algorithms for lattice quantizers and codes," *IEEE Trans. Inform. Theory*, vol. IT-28, pp. 227–232, Mar. 1982.

An Adaptive Morphological Filter for Image Processing

F. Cheng and A. N. Venetsanopoulos

***Abstract*—In this paper, a new type of opening operators (NOP) and closing operators (NCP) is proposed. An adaptive morphological filter is then constructed on the basis of the NOP and NCP. The filter can remove any details consisting of fewer pixels than a given number N, while preserving the other details. Efficient algorithms are also developed for the implementation of the NOP and NCP.**

I. Introduction

In recent years, morphological filters based on erosion, dilation, opening, and closing operations have attracted a lot of attention [1], [2]. The morphological operators of the early form have the drawbacks of creating artificial patterns, and distorting or removing significant details, as reported by many researchers [3]. Many approaches, such as the max (min) of a family of openings (closings) [4], and a new type of morphological operators based on the statistics of images [5], etc., have been considered to deal with those problems. Although improvements in performance have been reported with the use of these approaches, they may still not be quite satisfactory, for example, in the cases where the processed gray scale images may contain many patterns of significant details, especially in those cases where large structuring elements are required. To cope with the problems, we propose a new type of opening operators (NOP) and closing operators (NCP). An adaptive morphological filter is then developed on the basis of the NOP and NCP. The filter can preserve the significant details in any shape, does not create artificial patterns, and does not require much computation.

II. Conventional Morphological Operators

In this section, we give a brief description of the conventional morphological operators as the basis to develop the NOP and NCP in the next section. Let Z denote the set of integers and $x(i, j)$ denote a discrete image signal, where the domain set $\{i, j\} \subset Z^2$ and the range set $\{x\} \subset Z$. Throughout this paper, we only consider rectangular sampling grid and flat structuring elements. Hence, a structuring element can be expressed by its support domain $B \subset Z^2$. Denote $B^S = \{-b: b \in B\}$ as the symmetric set of B and B_{t_1,t_2} as the translation of B by (t_1, t_2), where $(t_1, t_2) \in Z^2$. The erosion $x \ominus B^S$ and dilation $x \oplus B^S$ can be expressed as [1]

$$(x \ominus B^S)(i, j) = \min_{(t_1,t_2)\in B_{i,j}} (x(t_1, t_2)), \tag{1}$$

$$(x \oplus B^S)(i, j) = \max_{(t_1,t_2)\in B_{i,j}} (x(t_1, t_2)). \tag{2}$$

Manuscript received October 16, 1990; revised May 28, 1992.
The authors are with the Department of Electrical Engineering, University of Toronto, Toronto, Ont., Canada M5S 1A4.
IEEE Log Number 9202394.

Opening $x \bigcirc B$ is defined as [1]

$$(x \bigcirc B)(i, j) \triangleq ((x \ominus B^S) \oplus B)(i, j) = \max_{((s_1,s_2):i,j\in B_{s_1,s_2})} [\min_{t_1,t_2\in B_{s_1,s_2}} (x(t_1, t_2))]. \tag{3}$$

Let G denote a group of structuring elements. Then the max of the openings, whose structuring elements are in G, is defined as

$$(x \bigcirc G)(i, j) \triangleq \max_{B^{(k)}\in G} [(x \bigcirc B^{(k)})(i, j)]. \tag{4}$$

Limited by computational complexity, in practice, G usually contains only a few structuring elements. Hence, the ability of the max of openings described by (4) in preserving details and suppressing artificial patterns is also limited since a natural image may contain far more than just a few patterns of significant details. By the duality between opening and closing, the min of closings can be described in a similar way, and has the same limitation. In fact, that limitation is shared by many other nonlinear filters that combine a group of windows [3].

III. The NOP and NCP

In this section, we first deal with the case of the NOP, and then extend the results to the case of the NCP. Let $\boldsymbol{B}^N$ denote the set of the structuring elements of all the shapes formed by N eight-connected pixels. Those structuring elements, which differ only by a translation, are considered identical and are taken as one element in $\boldsymbol{B}^N$, since the position of the structuring element does not affect the result of opening. Then, the proposed NOP $(x \bigcirc \boldsymbol{B}^N)$ is defined as

$$(x \bigcirc \boldsymbol{B}^N)(i, j) \triangleq \max_{B^{(k)}\in \boldsymbol{B}^N} [(x \bigcirc B^{(k)})(i, j)]. \tag{5}$$

On the basis of the properties of the max of openings [4], the NOP $(x \bigcirc \boldsymbol{B}^N)$ is an operator without shape bias. That is, it removes the bright objects smaller than a given threshold, and preserves those of any shape, whose sizes are not smaller than the threshold. In practice, there are a lot of cases where the operators without shape bias are found useful. For example: 1) in noise filtering, the operators can preserve the significant details in any shape, and will not create artificial patterns, and 2) based on the property of the visual system, in an image, we can roughly consider large objects more important and the objects smaller than some threshold less important. The operators can be used to decompose images based on that difference for various applications.

In the definition of the NOP, we only considered connected structuring elements. The reason is that this paper is a part of our effort to develop new nonlinear operators for image processing based on geometric structures in images. At present, we are only able to describe the connected objects as geometrical structures. Hence, the NOP is developed in a corresponding way. To describe the clustered pixels as geometrical structures poses a more challenging problem. We consider that, only after the problem is solved, it is possible to systematically develop the adaptive morphological operators, whose structuring elements are not necessary to be connected.

Through the work of Matheron and Serra, it has been known that the max (min) of a family of openings (closings) is an algebraic opening (closing) [7]. However, no one has suggested the NOP $(x \bigcirc \boldsymbol{B}^N)$, and its application because of the intolerable computational burden of $(x \bigcirc \boldsymbol{B}^N)$ by direct computation. The computational burden of $(x \bigcirc \boldsymbol{B}^N)$ by direct computation can be explained by the very fast increase of the number of elements in $\boldsymbol{B}^N$ with respect to N. For example, $\boldsymbol{B}^2$ has 4 elements and $\boldsymbol{B}^3$ has 20. It soon becomes difficult to enumerate all the elements of $\boldsymbol{B}^4$, perhaps several hundreds of them. Generally, it is a very hard task to construct all the elements of $\boldsymbol{B}^N$. Hence, we can say that without a new method, the practical computation of $(x \bigcirc \boldsymbol{B}^N)$ is impossible.

In order to develop a new method, we consider what is really performed by $(x \bigcirc \boldsymbol{B}^N)$ in (5). By combining with (3), (5) can be expressed as

$$(x \bigcirc \boldsymbol{B}^N)(i, j) = \max_{B^{(k)} \in \boldsymbol{B}^N} \Big[\max_{((s_1, s_2): i, j \in B^{(k)}_{s_1, s_2})} \Big[\min_{t_1, t_2 \in B^{(k)}_{s_1, s_2}} (x(t_1, t_2)) \Big] \Big]. \quad (6)$$

In (6), the minimum of the input signal x is computed in the domain of every translation $B^{(k)}_{s_1, s_2}$, which contains (i, j), of every structuring element $B^{(k)}$ in $\boldsymbol{B}^N$. Then the maximum is computed over all the minima obtained. In other words, we consider all the domains containing (i, j) and formed by N connected pixels. We assign to the NOP $(x \bigcirc \boldsymbol{B}^N)(i, j)$ the minimum of x in such a domain that the minimum of x in any other domain will not be larger than the value assigned to $(x \bigcirc \boldsymbol{B}^N)(i, j)$. Those results are summarized in Proposition 1.

Proposition 1: An equivalent definition of the NOP $(x \bigcirc \boldsymbol{B}^N)$ defined in (5) can be expressed in the following.

1) Search for N connected pixels containing (i, j), on which the minimum of x is not smaller than the minimum of x on any other N connected pixels containing (i, j).
2) Assign the minimum to $(x \bigcirc \boldsymbol{B}^N)(i, j)$.

According to Proposition 1, in order to compute the NOP, we only have to search $N - 1$ pixels connected to (i, j) along a path, which corresponds to the $N - 1$ largest values of x. The N connected pixels searched are in fact the points in the structuring element of the opening in (5) that gives the maximum value, i.e.:

$$\Big(\max_{B^{(k)} \in \boldsymbol{B}^N} [(x \bigcirc B^{(k)})(i, j)]\Big).$$

With conventional approaches, we have to compute all the openings in (5) and then compute the maximum. By contrast, Proposition 1 gives a way to directly find the right opening. All the other openings in (5) do not have to be computed. In that way, Proposition 1 offers a great potential for developing fast algorithms to compute $(x \bigcirc \boldsymbol{B}^N)$, and thus makes the study of $(x \bigcirc \boldsymbol{B}^N)$ meaningful.

By the duality between opening and closing, the cause of the NCP can be described in a similar way. We omit the details and only briefly mention the results. The NCP $(x \bullet \boldsymbol{B}^N)$ is defined as

$$(x \bullet \boldsymbol{B}^N)(i, j) \triangleq \min_{B^{(k)} \in \boldsymbol{B}^N} [(x \bullet B^{(k)})(i, j)]. \quad (7)$$

Proposition 2 gives an equivalent description of (7).

Proposition 2: An equivalent definition of the NCP $(x \bullet \boldsymbol{B}^N)$ defined by (7) can be expressed as follows.

1) Search for N connected pixels containing (i, j), on which the maximum of x is not larger than the maximum of x on any other N connected pixels containing (i, j).
2) Assign the maximum to $(x \bullet \boldsymbol{B}^N)(i, j)$.

IV. The Algorithms for the Computation of the NOP and NCP

A. The Basic Algorithm

The basic algorithm is a direct realization of Proposition 1, without further considering the properties of the NOP $(x \bigcirc \boldsymbol{B}^N)$. We first discuss the details of the case of $N \leq 9$, then briefly describe the case of $N > 9$. According to Proposition 1, the computation of $(x \bigcirc \boldsymbol{B}^N)(i, j)$ is basically a search for a domain described in step 1) of Proposition 1. Denote the domain by $S_{i,j}$. At the beginning of the search, $S_{i,j}$ contains one pixel (i, j). After the search is completed, $S_{i,j}$ contains N connected pixels including (i, j), on which x has the largest values. The basic algorithm has three steps. Step 0 is initialization. Step 1 is a search at the neighbors of the pixels included in $S_{i,j}$ for the candidates considered in Step 2. In Step 2, we determine which pixels picked up in Step 1 can be included in $S_{i,j}$ according to the conditions in Proposition 1. If the number of the pixels included in $S_{i,j}$ is still less than N, we go back to step 1 again. Steps 1 and 2 constitute a search cycle.

Before describing the search procedure, we define a number of buffers and the corresponding counters.

1) $S_{i,j}$ contains the pixels included so far. Counter M indicates how many pixels are in $S_{i,j}$. We use x_0 to denote the minimum value of x at the pixels included in $S_{i,j}$ so far.
2) Buffer $\{\underline{\mathbf{a}}_n\}$ contains the pixels included in $S_{i,j}$ during the previous search cycle. The corresponding pixel counter is AN.
3) Buffer $\{\underline{\mathbf{b}}_n\}$ contains the pixels, which are candidates, but are not included in $S_{i,j}$ in the previous search cycle. BN is the counter.
4) Buffer $\{\underline{\mathbf{c}}_n\}$ contains the neighboring pixels of $\{\underline{\mathbf{a}}_n\}$. CN is the counter.
5) Buffer $\{\underline{\mathbf{d}}_n\}$ contains $N - M$ pixels chosen from $\{\underline{\mathbf{b}}_n\}$ and $\{\underline{\mathbf{c}}_n\}$, which correspond to the $N - M$ largest values of x at the pixels in $\{\underline{\mathbf{b}}_n\}$ and $\{\underline{\mathbf{c}}_n\}$. Pixels in $\{\underline{\mathbf{d}}_n\}$ are candidates in the current search cycle. DN is the counter.

The following is the search procedure.

Step 0: Pixel (i, j) is included in $S_{i,j}$.
Assign the initial values:
$S_{i,j}$ contains 1 pixel. Thus assign $1 \rightarrow M$. Assign $x(i, j) \rightarrow x_0$.
Assign $(i, j) \rightarrow \underline{\mathbf{a}}_1$. $\{\underline{\mathbf{a}}_i\}$ contains 1 pixel. Thus assign $1 \rightarrow AN$.
$\{\underline{\mathbf{b}}_i\}$, $\{\underline{\mathbf{c}}_i\}$, and $\{\underline{\mathbf{d}}_i\}$ are empty. Thus assign $0 \rightarrow BN$, CN, and DN.

Step 1: Assign the neighboring pixels of $\underline{\mathbf{a}}_i$, $i = 1, \ldots, AN$ to $\underline{\mathbf{c}}_i$. Use a checkboard to record which pixel has been searched to avoid pick up of the pixels previously searched in the computation at (i, j). Assign the number of the pixels in set $\{\underline{\mathbf{c}}_i\}$ to counter CN.

Choose the $N - M$ largest values from $x(\underline{\mathbf{b}}_i)$ and $x(\underline{\mathbf{c}}_j)$, $i = 1, \cdots, BN$, $j = 1, \cdots, CN$. Order the $N - M$ values and assign the corresponding $N - M$ pixels to $\underline{\mathbf{d}}_i$ in the order that $x(\underline{\mathbf{d}}_i) \geq x(\underline{\mathbf{d}}_{i+1})$. Assign $N - M \rightarrow DN$.

Step 2: Comparing $x(\underline{\mathbf{d}}_i)$ with x_0, we have three cases.

1) If $x(\underline{\mathbf{d}}_1) < x_0$, then assign $\mathbf{x}(\underline{\mathbf{d}}_1) \rightarrow x_0$, $\underline{\mathbf{d}}_1 \rightarrow \underline{\mathbf{a}}_1$, $1 \rightarrow AN$; $\underline{\mathbf{d}}_{i+1} \rightarrow \underline{\mathbf{b}}_i$, for $i = 1, \cdots, DN - 1$; $DN - 1 \rightarrow BN$; and $M + 1 \rightarrow M$.
If $M = N$, assign $x_0 \rightarrow (x \bigcirc \boldsymbol{B}^N)(i, j)$, quit.
If $M < N$, goto Step 1.
2) If $x(\underline{\mathbf{d}}_{DN}) \geq x_0$, then assign $x_0 \rightarrow (x \bigcirc \boldsymbol{B}^N)(i, j)$, quit.
3) If $x(\underline{\mathbf{d}}_k) \geq x_0$, $x(\underline{\mathbf{d}}_{k+1}) < x_0$, $1 < k < DN$, assign $\underline{\mathbf{d}}_i \rightarrow \underline{\mathbf{a}}_i$, for $i = 1, \cdots, k$; $k \rightarrow AN$; $\underline{\mathbf{d}}_{i+k} \rightarrow \underline{\mathbf{b}}_i$, for $i = 1, \cdots, DN - k \rightarrow BN$; and $M + k \rightarrow M$. Then goto Step 1.

A brief explanation of the basic algorithm is given in the following.

1) In Step 1, we choose the $N - M$ largest values from $x(\underline{\mathbf{b}}_j)$ and $x(\underline{\mathbf{c}}_i)$. In this way, the max requirement in Proposition 1 is satisfied.

2) In 1) of Step 2, $\underline{\mathbf{d}}_1$ has to be included in $S_{i,j}$ since $\underline{\mathbf{d}}_i$, $i = 1, \cdots, DN$ are the only pixels to connect the pixels in $S_{i,j}$ with the pixels not yet searched. On the other hand, we only have to include $\underline{\mathbf{d}}_1$ in $S_{i,j}$ since there is a possibility for the values of x at the neighboring pixels of $\underline{\mathbf{d}}_1$ to be larger than $x(\underline{\mathbf{d}}_i)$, $i = 2, \cdots, DN$.
3) In 2) and 3) of Step 2, any $\underline{\mathbf{d}}_i$ satisfying $x(\underline{\mathbf{d}}_i) \geq x_0$ can be included in $S_{i,j}$ since including $\underline{\mathbf{d}}_i$ will not reduce x_0. Thus the max requirement of Proposition 1 is satisfied.
4) Step 2 shows that in each search cycle, we can include at least one pixel in $S_{i,j}$. Hence, the computation of the NOP at any pixel will have less than N search cycles.

In the case of $N > 9$, in Step 1 of the first search cycle, $CN = 8$, $BN = 0$. Hence, there are no $N - 1$ pixels to assign to $\underline{\mathbf{d}}_i$. To overcome this difficulty, we only have to modify the number $N - M$ in Step 1 in the first several search cycles.

The boundary pixels of an image can be handled by embedding the image into a one pixel wide frame for any size N of the structuring element. In the case of the NOP, the value at the pixels of the frame is assigned the minimum value of the image. Since the search in Step 1 of the basic algorithm is to find the pixels corresponding to the $N - M$ largest values of x, the minimum value guarantees that the search will not go beyond the frame.

A simple example is given to illustrate the effect of the NOP ($x \bigcirc \boldsymbol{B}^N$) on an image surface. The image surface is shown in Fig. 1. Assume the uniform area of the tip on the surface is smaller than N pixels. Then, the NOP ($x \bigcirc \boldsymbol{B}^N$) flattens the tip. The shape of the flattened tip is the shape of the structuring element of the NOP at that location, and it can be any shape of N pixels. At any pixel (s,t) on the slope and bottom parts of the surface, we can always find $N - 1$ pixels connected to (s,t), at which x is not smaller than $x(s,t)$. Hence, according to 2) of Step 2 in the basic algorithm, $x(s,t)$ will not be changed by the NOP. A complete description of the geometric performance of the NOP requires more understanding of the properties of the NOP. Those properties are currently under investigation.

B. Techniques for the Development of Fast Algorithms

The basic algorithm can be modified on the basis of the properties of the NOP ($x \bigcirc \boldsymbol{B}^N$) to develop fast algorithms. In this section, we describe two of these properties.

Proposition 3: In Step 1 of the basic algorithm, if $x(\underline{\mathbf{d}}_1) = x(\underline{\mathbf{d}}_{DN})$, then

$$(x \bigcirc \boldsymbol{B}^N)\,(i, j) = \min\,[x_0, x\,(\underline{\mathbf{d}}_{DN})]. \quad (8)$$

Proposition 3 can be proved by considering that $N = M + DN$, and that $\underline{\mathbf{d}}_i$, $i = 1, \cdots, DN$, are the only pixels connecting the pixels already included in $S_{i,j}$ with the not yet searched pixels, at which x may be larger than $x(\underline{\mathbf{d}}_{DN})$.

Proposition 4: If the values of x at the pixels, which are the neighbors of $\underline{\mathbf{d}}_i$, $i = 1, \cdots, DN$, and which are not included in $S_{i,j}$, are smaller than $x(\underline{\mathbf{d}}_{DN})$, then

$$(x \bigcirc \boldsymbol{B}^N)\,(i, j) = \min\,[x_0, x\,(\underline{\mathbf{d}}_{DN})]. \quad (9)$$

The proof can be based on the same consideration as that for the proof of Proposition 3. Since it is relatively straight forward to combine Propositions 3 and 4 into the basic algorithm, we omit the related explanations.

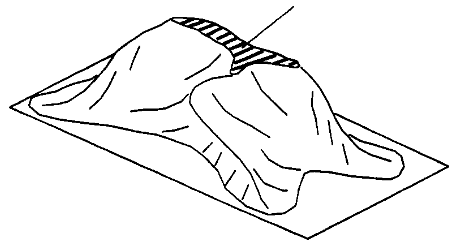

Fig. 1. Geometric performance of NOP.

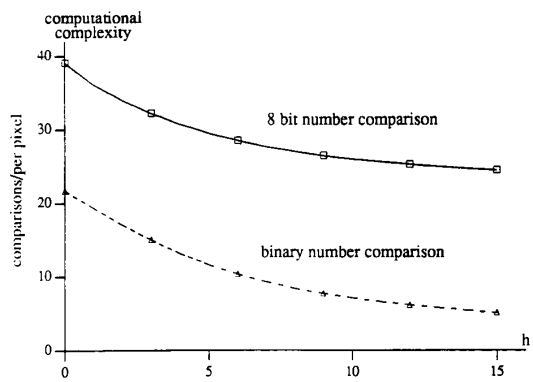

Fig. 2. The effect of h on computational complexity.

C. The Computational Complexity of the Algorithms

The computational complexity of such a type of algorithms is difficult to analyze because of the following two facts.

1) The computational complexity heavily depends on the complexity of the geometric structures of images. There is no suitable model to represent a nature image on a geometric basis.
2) Statistic models of images do not work in this case. For example, if we consider a Gaussian distribution of $x(i, j)$, then it will be very hard to know the distribution of x in $S_{i,j}$.

Generally, the computational complexity can only be measured by setting a counter in the algorithm.

In this paper, in order to further reduce computational complexity, we experimentally introduced a threshold, h, in our algorithm so that the differences smaller than h are ignored in the comparison of the values of x. The introduction of h makes the operators no longer openings, but our experiments show that when $h \leq 5$, there is no visible difference between the images processed by the NOP and by the NOP with h. By that meaning, for convenience, we still call the NOP with h the NOP in all of our work. For the same reason as those mentioned at the beginning of this section, it is difficult to analyze how h affects the computational complexity of the algorithms. The effect of h on the computational complexity is only experimentally shown in Fig. 2.

In Table I, the number of comparisons/per pixel of the NOP algorithm designed in this paper is compared with that of the conventional algorithm of the max of four openings. In the table, 8b and 1b indicate 8 bit and binary number comparisons, respectively. Since the computational complexity of the NOP algorithm changes

TABLE I
NUMBER OF COMPARISONS/PER PIXEL

Size N (points)	NOP Algorithm				Opening with 4 1-D Elements
	Noisy Lena		Harbor		
	8 b	1 b	8 b	1 b	8 b
3	11.5	0.24	11.5	0.37	19
4	14.8	0.79	15.5	1.81	27
5	18.8	2.10	20.3	4.13	35
6	23.7	5.51	26.1	8.47	43
7	29.4	11.6	33.0	15.7	51
8	35.6	18.7	39.5	22.2	59
9	43.8	30.8	47.1	31.6	67

with the complexity of the processed images, two test images are used to show the difference. One is "Lena" corrupted with 10% impulsive noise, while the other is "Harbor" which is noise free. The purpose of the Table I is to give an idea of the computational complexity of the NOP algorithm by comparing it with a commonly used algorithm. As to the ability in computing $(x \bigcirc \boldsymbol{B}^N)$, there is no comparison of the conventional approaches with the approach proposed in this paper, since as mentioned in Section III, $(x \bigcirc \boldsymbol{B}^N)$ cannot be computed by the conventional approaches. Although the number of the comparisons/per pixel of the NOP algorithm in Table I is impressively low considering the complexity of the task, our work on the properties of the NOP has shown that it is still possible to develop much faster algorithms.

D. The Extension to the Case of the NCP

By the duality between closing and opening, the results obtained in Section IV-A-C can be easily extended to the case of the NCP. For example, the extension of the basic algorithm only requires the inverse of the ordering computation, and the change of the directions of the inequalities related to the comparison of x.

V. THE ADAPTIVE MORPHOLOGICAL FILTER

The NOP and NCP in this paper are still in their initial simplest form. Our investigation has revealed many possible applications of the modified forms of the NOP and NCP. The modification of the NOP and NCP is not the topic of this paper. Here, as an application example, we consider the removal of impulsive noise, since in this case, even the simplest form of the NOP and NCP has shown significant improvement over many existing techniques.

The adaptive morphological filter consists of two parts. The main part of the filter is an NOP and an NCP. The size of the structuring element is determined according to the image patterns and the impulsive noise patterns. The second part is a postprocessing. According to Proposition 1, a noise pattern connected to the edge of a large object will be counted as part of the large object by the NOP and NCP, and thus will not be completely removed by the first part of filtering. Hence, a postprocessing is required to remove such noise patterns. Among many possible ways of postprocessing, in this paper, we only consider a very simple way. Let $y(i, j)$ denote the image already processed by the NOP and NCP. The main idea of the proposed postprocessing is the following.

1) To decompose y into z_1 and z_2 by a conventional opening-closing filter with a small (for example, 2 by 2 pixels) structuring element B to separate the remaining noise patterns from large objects.

$$z_1(i, j) = (y \bigcirc \boldsymbol{B}) \bullet \boldsymbol{B}; \quad z_2(i, j) = y(i, j) - z_1(i, j). \tag{10}$$

2) To remove the noise patterns from $z_2(i, j)$ by the NOP and NCP with a structuring element of a small size N_1 (for example 3 or 4 pixels).

$$z_3(i, j) = (z_2 \bigcirc \boldsymbol{B}^{N_1}) \bullet \boldsymbol{B}^{N_1}. \tag{11}$$

3) The final image is $z_1(i, j) + z_3(i, j)$.

VI. THE EXPERIMENTAL RESULTS

In this section, we compare the proposed adaptive morphological filter with the signal adaptive median filter [6], and with the max of openings and the min of closings, which combines four 1-D structuring elements oriented at the angles of 0, 45, 90, and 135°.

The filters are compared by subjective criterion on the processed images and by the normalized mean square error (NMSE) defined by NMSE $= [\Sigma_{i,j} (x(i, j) - y(i, j))^2]/[\Sigma_{i,j} x(i, j)^2]$, where $x(i, j)$ is the original image and $y(i, j)$ is the filtered image. All the images used in the experiments are of size 256 by 256. The original image "Lena" is shown in Fig. 3(a). The test image is "Lena" contaminated by 15% impulsive noise of random values, and is shown in Fig. 3(b). The NMSE is 6.16×10^{-2}. Fig. 3(c) shows the image processed by the signal adaptive median filter. The NMSE is 3.60×10^{-2}. The filter has failed to detect and remove many of the noise impulses since the values of the noise impulses are not significantly different from that of the image. Fig. 3(d) is the image processed by the max of openings and the min of closings with $N = 5$. The NMSE is 5.93×10^{-3}. By comparing with "Lena," we observe that the loss and distortion of the significant details are obvious. Artificial patterns are created in many parts of the detailed area. Fig. 3(e) is the image processed by the adaptive morphological filter with $N = 9$. The two sizes of the structuring elements are just enough for the corresponding filters to remove 15% impulsive noise of random values. The NMSE is 3.33×10^{-3}. Although the improvement in NMSE is not quite impressive, the improvement in visual perception is noticeable. First, the significant details of one pixel wide patterns in various shapes are almost all preserved. Secondly, no artificial patterns are created.

ACKNOWLEDGMENT

The authors wish to thank R. L. J. Martens in the same research group and the reviewers of this paper. R. L. J. Martens offered the images processed with the signal adaptive median filter. The comments of the reviewers helped to improve this paper.

(a)

(b)

Fig. 3. (a) Original image "Lena". (b) Noisy image. Images processed by (c) signal adaptive median filter. (d) Max of opening and min of closings. (e) The proposed filter.

(c)

(d)

Fig. 3. (*Continued*).

(e)

Fig. 3. (*Continued*).

REFERENCES

[1] J. Serra, *Image Analysis and Mathematical Morphology*. London, U.K.: Academic, 1982.

[2] P. Maragos and R. W. Schafer, "Morphological systems for multidimensional signal processing," *Proc. IEEE*, vol. 78, pp. 690–709, Apr. 1990.

[3] G. R. Arce and R. E. Foster, "Detail-preserving ranked-order based filters for image processing," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pp. 83–98, Jan. 1989.

[4] R. L. Stevenson and G. R. Arce, "Morphological filters: statistics and further syntactical properties," *IEEE Trans. Circuits Syst.*, vol. CAS-34, Nov. 1987.

[5] E. R. Dougherty, "Minimal search for the optimal mean-square digital grayscale morphological filter," in *Proc. of Visual Communications and Image Processing '90*, M. Kunt, ed., vol. 1360, 1990, pp. 214–225.

[6] R. Bernstein, "Adaptive nonlinear filters for simultaneous removal of different kinds of noise in images," *IEEE Trans. Circuits Syst.*, vol. CAS-34, pp. 1275–1291, Nov. 1987.

[7] G. Matheron, *Random Sets and Integral Geometry*. New York: Wiley, 1975.

Hidden Markov Models for Character Recognition

J. A. Vlontzos and S. Y. Kung

***Abstract*—Multifont and handwritten character recognition systems have not been successfully implemented to date because of the variability of characters and the difficulty of incorporating context in the classification process. Hierarchical systems are very useful in making context-based decisions but their knowledge sources cannot easily incorporate both general knowledge about objects and at the same time knowledge about object instantiations. In this paper we present a hierarchical system for character recognition with hidden Markov model knowledge sources which solve both the context sensitivity problem and the character instantiation problem. Our system achieves 97–99% accuracy using a two level architecture and has been implemented using a systolic array, thus permitting real time (1 ms per character) multifont and multisize printed character recognition as well as handwriting recognition.**

Manuscript received August 4, 1988; revised June 18, 1991. This work was supported in part by the Air Force Office of Scientific Research under Grant AFOSR-89-0501A.

J. A. Vlontzos is with Siemens Corporate Research, Cholargos 155-62, Greece.

S. Y. Kung is with Princeton University, Princeton, NJ 08540.

IEEE Log Number 9202151.

I. INTRODUCTION

Character recognition has attracted the attention of researchers since the early days of computing and a number of successful commercial systems have been produced. The existing systems, however, exhibit high recognition rates (99.9%) only for one font and are very sensitive to page misalignments [8]. Most systems use some form of template matching techniques to find the best match with a set of predetermined templates. Our goal in this paper is first to develop a system for multifont printed character recognition and then extend the ideas developed for handwritten character recognition.

In this paper we present a system for character recognition, largely inspired by multilevel recognition systems such as the HEARSAY system [1] for speech recognition. The advantage of such a system is, apart from permitting parallelism, that it can easily incorporate *context* in the form of knowledge about larger scale structures such as words and sentences.