/22

COMPUTER VISION, GRAPHICS, AND IMAGE PROCESSING **35**, 370–382 (1986)

Morphological Structuring Element Decomposition

XINHUA ZHUANG* AND ROBERT M. HARALICK

Machine Vision International, Ann Arbor, Michigan 48104

Received March 20, 1985; accepted April 10, 1986

To efficiently perform morphological operations with specialized pipeline hardware which is not able to utilize all the points in the domain of the structuring element in one pipeline stage requires the capability of decomposing the structuring element into a morphological dilation of smaller structuring elements each of which is utilized in a successive stage of the pipeline. In this paper, we give the theory and algorithm for such optimal structuring element decomposition. © 1986 Academic Press, Inc.

I. INTRODUCTION

Mathematical morphological operations [1] such as dilation, erosion, opening, and closing are important techniques in image processing. Specialized hardware such as the cytocomputer and the image flow computer [2–4] have been built which are capable of efficiently performing morphological operations. The cytocomputer is a pipelineable machine which can process a 3×3 neighborhood morphological operation in each stage. The image flow computer is a pipelineable machine which can process a morphological operation on any pair of pixels in each stage.

If the given structuring element used in the morphological operation has a domain larger than the domain which the hardware can handle in one stage, the structuring element must be decomposed into smaller structuring elements which each are capable of being handled by one stage in the pipeline and whose morphological composition is the given structuring element. The decomposition problem is to determine a smallest sequence of small structuring elements whose morphological composition is the given structuring element.

To make the problem definition more precise requires some morphological definitions. Let E^K be K dimensional Euclidean space and $X \subseteq E^K$ and $B \subseteq E^K$. The *dilation* of X by structuring element B is denoted by $X \oplus B$ and is defined by

$$X \oplus B = \{ y | \text{for some } x \in X \text{ and } b \in B,\ y = x + b \}. \tag{1}$$

The *erosion* of X by structuring element B is denoted by $X \ominus B$ and is defined by

$$X \ominus B = \{ y | \text{for every } b \in B,\ y + b \in X \}. \tag{2}$$

The reader should beware that the above definition of $\ominus$ differs from that given by Serra [1] in that he requires $y - b \in X$ instead of $y + b \in X$.

Two important idempotent morphological operations are composed of dilations and erosions. A dilation followed by an erosion is called a closing. An erosion followed by a dilation is called an opening. See Serra [1] for details about how openings and closings are used.

* Visiting scientist from People's Republic of China.

It follows in a straightforward manner from these definitions that dilation is associative and commutative and that

$$(X \ominus B_1) \ominus B_2 = X \ominus (B_1 \oplus B_2).$$

Hence if the structuring element S has the decomposition

$$S = H_1 \oplus H_2 \oplus \cdots \oplus H_N$$

the dilation of X by S can be performed as

$$X \oplus S = X \oplus (H_1 \oplus H_2 \oplus \cdots \oplus H_N) = (((X \oplus H_1) \oplus H_2) \cdots) \oplus H_N.$$

The erosion of X by S can be performed as

$$X \ominus S = X \ominus (H_1 \oplus H_2 \oplus \cdots \oplus H_N) = (\cdots ((X \ominus H_1) \ominus H_2) \cdots) \ominus H_N.$$

The structuring element decomposition problem is given a structuring element S. Determine the smallest N and corresponding structuring elements $H_1, H_2, \ldots, H_N$ such that

$$S = H_1 \oplus H_2 \oplus \cdots \oplus H_N$$

where each H_n satisfies the smallness requirement of the hardware pipelineable stages.

To make this decomposition problem more concrete, consider the following example in 2-dimensional Euclidean space. We take the structuring element S to be a 7×5 rectangle whose corner pixels are missing. What is the most efficient decomposition of this structuring element into structuring elements which are each 2-point sets? One decomposition, not the most efficient, is given by

$$\begin{aligned} S = {} & \{(0,0),(0,1)\} \oplus \{(0,0),(0,1)\} \oplus \{(0,0),(1,1)\} \oplus \{(0,0),(1,0)\} \\ & \oplus \{(0,0),(1,0)\} \oplus \{(0,0),(1,-1)\} \oplus \{(0,0),(0,-1)\} \oplus \{(0,0),(0,-1)\}. \end{aligned}$$

The theory and algorithm developed in this paper enable us to develop answers to questions like: Is the 8 2-point set decomposition given for S the smallest such decomposition? How is the smallest decomposition constructed?

In Section II, we detail the hardware motivation for using structuring elements which are highly decomposable. In Section III we initially describe a brute force search procedure to determine an optimal decomposition. The body of Section III discusses how to make the search a finite search which utilizes the forward checking technique [6] for reducing the number of possibilities actually searched. In Section IV are some examples of some optimal decompositions. Section V is the conclusion.

II. HARDWARE MOTIVATION FOR USING STRUCTURING ELEMENTS HAVING DECOMPOSITIONS

If each H_n in the decomposition of S,

$$S = H_1 \oplus \cdots \oplus H_N,$$

is a 2-point set, then the resulting decomposition is said to be a 2-point decomposition. In this section, we provide the hardware motivation for using a 2-point set decomposition. A canonical 2-point decomposition involves only H_n's for which $0 \in H_n$, $n = 1, \ldots, N$. Canonical 2-point decompositions are important because dilation with such a 2-point set can be accomplished by a shift and an OR. Erosion can be accomplished by a shift and an AND. We now review enough mathematical morphology to illustrate these facts.

We begin by stating the relationships among dilation, erosion, set union, and set intersection. These relationships are all easily proven in a few steps using the definitions of the operations:

$$\begin{aligned}
A \ominus (B \oplus C) &= (A \ominus B) \ominus C \\
A \oplus (B \cup C) &= (A \oplus B) \cup (A \oplus C) \\
(A \cup B) \oplus C &= (A \oplus C) \cup (B \oplus C) \\
(A \cap B) \ominus C &= (A \ominus C) \cap (B \ominus C) \\
A \ominus (B \cup C) &= (A \ominus B) \cap (A \ominus C).
\end{aligned}$$

Next we need the definition of translate. Let X be a subset of E^K and t be an element of E^K. Then the translate of X by t is denoted by X_t and is defined by

$$X_t = \{ y \mid \text{for some } x \in X,\ y = x + t \}.$$

In this context we call t a shift and say that X has been translated by the shift t or that X has been shifted by t.

It follows immediately from the definition of dilation and erosion that

$$\begin{aligned}
X \oplus \{t\} &= X_t \\
X \ominus \{t\} &= X_{-t} \\
A \oplus B_t &= (A \oplus B)_t \\
A \ominus B_t &= (A \ominus B)_{-t} = A_{-t} \ominus B.
\end{aligned}$$

Since $A \oplus (B \cup C) = (A \oplus B) \cup (A \oplus C)$ it follows directly that

$$A \oplus B = \bigcup_{b \in B} A_b.$$

This says that dilation can be accomplished by taking the union of all the translates of A where the shifts in the translates come from B. Since $A \ominus (B \cup C) = (A \ominus B) \cap (A \ominus C)$, it follows directly that

$$A \ominus B = \bigcap_{b \in B} A_{-b}.$$

This says that erosion can be accomplished by taking the intersection of all the translates of A, where the shifts in the translates are the negated members of B.

The hardware consequences for using canonical decompositions are important. If $H = \{0, h\}$, then

$$A \oplus H = A_0 \cup A_h = A \cup A_h$$
$$A \ominus H = A_0 \cap A_{-h} = A \cap A_{-h}.$$

Dilation with H is accomplished by a shift to produce A_h followed by a union with A. Erosion with H is accomplished by a shift to produce A_{-h} followed by an intersection with A.

To compute a dilation of A with a canonical two point decomposition $H_1 \oplus \cdots \oplus H_N$, the computation proceeds in the form

$$(\cdots((A \oplus H_1) \oplus H_2) \oplus \cdots) \cdots H_N$$

where each successive dilation is accomplished by taking the previous result, shifting it and ORing the shifted result with the previous result to produce the next result. That is, if $H_n = \{0, h_n\}$, $n = 1, \ldots, N$,

$$B^1 = A \oplus H_1 = A \cup A_{h_1}$$

and

$$B^n = B^{n-1} \oplus H_{n-1} = B^{n-1} \cup (B^{n-1})_{h_{n-1}}, \qquad n = 2, \ldots, N,$$

then the desired result $A \oplus H_1 \oplus \cdots \oplus H_N = B^N$.

Similarly, to compute an erosion of A with a canonical 2-point decomposition $H_1 \oplus \cdots \oplus H_N$, the computation proceeds in the form

$$(\cdots((A \ominus H_1) \ominus H_2) \ominus \cdots) \ominus H_N$$

where each successive erosion is accomplished by taking the previous result shifting it and ANDing the shifted result with the previous result to produce the next result. That is, if

$$B^1 = A \ominus H_1 = A \cap A_{-h_1}$$

and

$$B^n = B^{n-1} \ominus H_{n-1} = B^{n-1} \cap (B^{n-1})_{-h_{n-1}}, \qquad n = 2, \ldots, N$$

then the desired result $A \ominus (\mathrm{H}_1 \oplus \cdots \oplus H_N) = B^N$.

III. SEARCHING FOR THE DECOMPOSITION

III. 1. Overview

A structuring element S is decomposable into the N structuring elements $H_1, \ldots, H_N$ if and only if $S = H_1 \oplus \cdots \oplus H_N$. The problem we solve in this section is to construct a decomposition of S having the smallest N, if one exists, where each H_n has no more members than the prescribed fixed number k determined by hardware constraints. We also assume that S is finite in size.

To determine such a decomposition of S, if one exists, requires a combinatorial search process. The search begins at the root node. Consider an m-level mode. It contains the partial decomposition $H_1 \oplus \cdots \oplus H_m$. If there is no translation t of $H_1 \oplus \cdots \oplus H_m$ such that $(H_1 \oplus \cdots \oplus H_m)_t \subseteq S$ then the node dies. If there is a translation which is a subset of S, then the node lives. Suppose that there are M possible structuring elements which can be considered for H_{m+1}. Then this m-level node has M children. If at some level in the tree no node survives, the tree search terminates and S has no decomposition. If S has an optimal decomposition (a decomposition with the smallest N), then one will be found by a breadth-first search. It will be the first one found by a breadth-first search.

To help make this search be more efficient we show how to search over a form equivalent to $H_1 \oplus \cdots \oplus H_N$, but one with fewer degrees of freedom. We establish that if $S = H_1 \oplus \cdots \oplus H_N$, then there exists a q and $J_1, \ldots, J_N$ such that

(1) $\#J_n = \#H_n$, $n = 1, \ldots, N$

(2) $0 \in J_n$, $n = 1, \ldots, N$

(3) $S = \{q\} \oplus J_1 \oplus \cdots \oplus J_N$

(4) $\{q\} = S \ominus (J_1 \oplus \cdots \oplus J_N)$.

Since $\#J_n = \#H_n$, $n = 1, \ldots, N$, the number of unknowns required to determine each J_n is the same as that required to find each H_n. The equivalent search is not more difficult than the original search. However, since it is known that $0 \in J_n$, $n = 1, \ldots, N$, there is one less unknown element to search over to determine each J_n. Thus the search to determine $J_1, \ldots, J_N$ will be less complex than the search to determine $H_1, \ldots, H_N$. Since $\{q\} = S \ominus (J_1 \oplus \cdots \oplus J_N)$, the added unknown q is determined without search once $J_1, \ldots, J_N$ have been determined.

To assure a finite or terminating search, we show that the kinds of elements each J_n can contain are limited to differences between elements of S. Thus if S is finite, the number of possibilities for each element of J_n is finite.

Finally, we establish that the search required to determine $J_1, \ldots, J_N$ can be made more efficient by the employment of forward checking. Forward checking tree searches never instantiate a possibility if somewhere earlier in the tree search the possibility failed and the problem guarantees that if a possibility fails any place in the tree search then it must fail in each instance it is instantiated anywhere in the subtree below where it initially failed. In the case of the structuring element decomposition problem, we show that if at some m-level node of the tree search $J_1, \ldots, J_m$ have been determined, then the only J that need be considered for any node in the subtree below this node must be a J satisfying

$$(T \ominus J) \oplus J \oplus K = S$$

where

$$T = S \ominus K$$
$$K = J_1 \oplus \cdots \oplus J_m.$$

This happens because if S has the decomposition $S = \{q\} \oplus J_1 \oplus \cdots \oplus J_N$ then it necessarily follows that $S = [S \ominus (J_1 \oplus \cdots \oplus J_n)] \oplus (J_1 \oplus \cdots \oplus J_n)$, $n = 1, \ldots, N$.

Hence if for some n, $S \neq [S \ominus (J_1 \oplus \cdots \oplus J_n)] \oplus (J_1 \oplus \cdots \oplus J_n)$ all possible completions of the partial decomposition $J_1 \oplus \cdots \oplus J_n$ to the decomposition $J_1 \oplus \cdots \oplus J_N$ must fail.

III. 2. Details

First we establish that if $S = H_1 \oplus \cdots \oplus H_N$ then there is an equivalent decomposition $S = \{q\} \oplus J_1 \oplus \cdots \oplus J_N$, where $\#J_n = \#H_n$ and $0 \in J_n$, $n = 1, \ldots, N$, and $\{q\} = S \ominus (J_1 \oplus \cdots \oplus J_N)$. To do this we use the concept of a translate introduced in Section II.

To see why $S = H_1 \oplus \cdots \oplus H_N$ implies $S = \{q\} \oplus J_1 \oplus \cdots \oplus J_N$, select one t_n from each H_n, $n = 1, \ldots, N$. Define $J_n = (H_n)_{-t_n} = \{h \mid \text{for some } x \in H_n,\ h = x - t_n\}$. Since $t_n \in H_n$, it is obvious that J_n must contain the origin. Since each J_n is just the translate of H_n, $\#J_n = \#H_n$. Now consider $H_1 \oplus \cdots \oplus H_N$:

$$\begin{aligned} H_1 \oplus \cdots \oplus H_N &= (J_1)t_1 \oplus \cdots \oplus (J_N)_{t_N} \\ &= (J_1 \oplus \cdots \oplus J_N)t_1 + \cdots + t_N \\ &= (J_1 \oplus \cdots \oplus J_N)q = \{q\} \oplus J_1 \oplus \cdots \oplus J_N \end{aligned}$$

where $\{q\} = t_1 + \cdots + t_N$.

To prove that if $S = \{q\} \oplus J_1 \oplus \cdots \oplus J_N$ implies that $\{q\} = S \ominus (J_1 \oplus \cdots \oplus J_N)$ is more involved. First we establish in Propositions 1 and 2 that if H is non-empty and finite then $H_x \subseteq H$ implies $x = 0$. From this it will follow in Proposition 3 that $\{p\} = H_p \ominus H$. From there it follows immediately in Proposition 4 that $S = \{q\} \oplus J$ implies $\{q\} = S \ominus J$. Taking $J = J_1 \oplus \cdots \oplus J_N$, we have our result.

PROPOSITION 1. *$H_x \subseteq H$ for every x and $h \in H$ imply $h + kx \in H$ for every x and for every nonnegative integer k.*

Proof. When $k = 0$, $h + kx = h$ and we have $h \in H$ by assumption. Suppose $h + kx \in H$. We will show that $h + (k+1)x \in H$. If $h + kx \in H$ and $H_x \subseteq H$, then $(h + kx) + x \in H$. Hence $h + (k+1)x \in H$. By induction $h + kx \in H$ for every nonnegative integer k.

PROPOSITION 2. *Suppose H is non-empty and bounded. Then $H_x \subseteq H$ implies $x = 0$.*

Proof. Suppose $H_x \subseteq H$ and $x \neq 0$. Since H is non-empty there exists some $h \in H$. Now $h \in H$ and $H_x \subseteq H$ implies $h + kx \in H$ for every nonnegative integer k. But $\lim_{k \to \infty} h + kx = \infty$, which contradicts the boundedness of H. Hence $x = 0$.

PROPOSITION 3. *Suppose H is non-empty and bounded. Then $\{p\} = H_p \ominus H$.*

Proof. Let $x \in H_p \ominus H$. Then for every $h \in H$, $x + h \in H_p$. Hence $H_x \subseteq H_p$ or $H_{x-p} \subseteq H$. But H non-empty and bounded and $H_{x-p} \subseteq H$ imply $x - p = 0$, or $x = p$. Since x was an arbitrary point in $H_p \ominus H$, $H_p \ominus H = \{p\}$.

COROLLARY. *Suppose J is non-empty and bounded. Then $S = \{q\} \oplus J$ implies $\{q\} = S \ominus J$.*

Proof. $S = \{q\} \oplus J$ implies $S = J_q$. J non-empty and bounded implies $\{q\} = J_q \ominus J$. Since $S = J_q$, we have $\{q\} = S \ominus J$.

Next we establish that if $S = \{q\} \oplus J_1 \oplus \cdots \oplus J_N$, where $0 \in J_n$, $n = 1, \ldots, N$, then $j \in J_n$ implies there exists a $p_1 \in S$ and $p_2 \in S$ such that $j = p_1 - p_2$. Since dilation is commutative, we can without loss of generality consider an S of the form $S = A \oplus J$, where $0 \in J$, and show that each $j \in J$ consists of the difference between a pair of elements of S. The proof consists of two steps as indicated in Propositions 4, 5, and 6. Proposition 4 establishes that the opening $(S \ominus J) \oplus J$ is always contained in S. Proposition 5 establishes that if $S = A \oplus J$ then $S = (S \ominus J) \oplus J$. And Proposition 6 proves that if $S = (S \ominus J) \oplus J$ with $0 \in J$ then $j \in J$ implies that for some $p_1 \in S$ and $p_2 \in S$, $j = p_1 - p_2$.

DEFINITION. The opening of S by J is denoted by $S \circ J$ and is defined by $S \circ J = (S \ominus J) \oplus J$.

PROPOSITION 4. $S \circ J \subseteq S$.

Proof. Let $s \in S \circ J$. Then there exists an $x \in S \ominus J$ and $j \in J$ such that $s = x + j$. But $x \in S \ominus J$ implies that for every $y \in J$, $x + y \in S$. Since $j \in J$, $x + j \in S$. But $s = x + j$ so that $s \in S$.

PROPOSITION 5. *If* $S = A \oplus J$ *then* $S = S \circ J$.

Proof. Let $s \in S$. Since $S = A \oplus J$, there exists an $a \in A$ and $j \in J$ such that $s = a + j$. But for every $x \in A$ and $j \in J$, $x + j \in S$. Since $a \in A$, we must have for every $y \in J$, $a + y \in S$. This implies $a \in S \ominus J$. Now $s = a + j$ with $a \in S \ominus J$ and $j \in J$ imply $S \in (S \ominus J) \oplus J$. Hence $S \subseteq (S \ominus J) \oplus J$. By Proposition 4, $S \circ J \subseteq S$. Finally, $S = S \circ J$.

PROPOSITION 6. *Suppose* $S = S \circ J$ *and* $0 \in J$. *Then* $j \in J$ *implies that there exists a* $p_1 \in S$ *and* $p_2 \in S$ *such that* $j = p_1 - p_2$.

Proof. Let $j \in J$. By definition of dilation, if $S = S \circ J$ then there exists an $s \in S$ and $y \in S \ominus J$ such that $y + j = s$. Since $y \in S \ominus J$, for every $z \in J, y + z \in S$. But $0 \in J$. Hence, $y + 0 \in S$ so that $y \in S$. Thus $j = s - y$, where $s \in S$ and $y \in S$.

We can make immediate use of the fact that each structuring element consists of members which are differences between members of S. Suppose that S has m members and that each subscripted J must have k members, one of which is 0. Each member of J consists of the difference between a pair of members of S. There are $n = m(m-1)$ possible differences. From these n possibilities each J must choose $k - 1$ members, the kth member being 0. The order in which the members are chosen is not important. The number of different possibilities for J is then $M = n!/(n-k+1)!(k-1)!$. We designate these possibilities $J^1, \ldots, J^M$.

In the case of 2-point structuring elements, there are even less possibilities since it is not necessary to consider a structuring element and its translate. If $H = \{0, h\}$ is one structuring element which is generated by a difference, then its translate by $-h$, $H_{-h} = \{-h, 0\}$, is another one which will also be generated by a difference. Only one is really needed in the decomposition, since any decomposition which involved H_{-h} could be changed to a decomposition which involves H instead, just by transferring the translation to the q in the decomposition $S = \{q\} \oplus J_1 \oplus \cdots \oplus J_N$.

In the brute force tree search each node has M children consisting of precisely $J^1, \ldots, J^M$. However, since dilation is commutative, a brute force tree search would find a decomposition solution $J^{i_1} \oplus J^{i_2} \oplus \cdots \oplus J^{i_N}$ and independently find $N!$

more decomposition solutions, each of the form $J^{n_1} \oplus J^{n_2} \oplus \cdots \oplus J^{n_N}$ where $(n_1, n_2, \ldots, n_N)$ is a permutation of $(i_1, i_2, \ldots, i_N)$. This additional work is easily eliminated by a restricted tree search. If a node in the tree is associated with possibility J^q, then instead of having M children $J^1, \ldots, J^M$, it only need have $M - q + 1$ children $J^q, \ldots, J^M$. It is obvious that this restricted tree search does not miss any decomposition. Suppose, for example, that $J^{i_1} \oplus \cdots \oplus J^{i_N}$ is a decomposition. Then there is some permutation $(n_1, \ldots, n_N)$ of $(i_1, \ldots, i_N)$ satisfying $n_1 \leq n_2 \leq \cdots \leq n_N$. Thus the restricted tree search will find the decomposition $J^{n_1} \oplus \cdots \oplus J^{n_N}$ which is equal to $J^{i_1} \oplus \cdots \oplus J^{i_N}$, since dilation is commutative. It is also obvious that any two decompositions the restricted tree search produces must be different for each permutation of $(i_1, \ldots, i_N)$ can only occur once in a representation constrained so that $i_1 \leq i_2 \leq \cdots \leq i_N$.

Consider the state of affairs at a level $m + 1$ node of the tree search. Here $J_1, \ldots, J_m$ have already been determined. Let $K_m = J_1 \oplus \cdots \oplus J_m$. K_m represents the partial decomposition of S determined for the branch of the tree search terminating at the level m node. Let $T_m = S \ominus K_m$. T_m represents that part of S which is yet to be decomposed.

We assume that S has the form $S = \{q\} \oplus J_1 \oplus \cdots \oplus J_N$, $N > m$, and we need to determine a possible value J for J_{m+1}. Rewriting the form of S as

$$S = (\{q\} \oplus J_{m+2} \oplus \cdots \oplus J_N) \oplus K_m \oplus J_{m+1}$$

we see that by Proposition 5, a value J for J_{m+1} must satisfy

$$\begin{aligned} S &= (S \ominus (K_m \oplus J)) \oplus (K_m \oplus J) \\ &= (T_m \ominus J) \oplus J \oplus K_m. \end{aligned}$$

S must be open under $J \oplus K_m$ (i.e., the opening of S by $J \oplus K_m$ must equal S), since any J for which

$$S \neq (T_m \ominus J) \oplus J \oplus K_m$$

implies that there can exist no decomposition of S at this level m node involving $J \oplus K_m$.

Not only is this true for the level m node under consideration, but it is also true at every node in the subtree below this level m node. To see this let $n > m$. Consider $(T_n \ominus J) \oplus J \oplus K_n$:

$$\begin{aligned} (T_n \ominus J) \oplus J \oplus K_n &= ((S \ominus K_n) \ominus J) \oplus J \oplus K_n \\ &= (S \ominus (K_n \oplus J)) \oplus (K_n \oplus J). \end{aligned}$$

For J to be instantiated as a child of this level n node, it must satisfy

$$(S \ominus (K_n \oplus J)) \oplus (K_n \oplus J) = S \circ (K_n \oplus J) = S.$$

Now $K_n \oplus J = J_1 \oplus \cdots \oplus J_n \oplus J = (K_m \oplus J) \oplus (J_{m+1} \oplus \cdots \oplus J_n)$, and we can prove that if S is open under a dilation of two structuring elements, then S is open under either of the structuring elements. We establish this first by a characterization of opening in Proposition 7 from which the stated result follows almost immediately in Proposition 8. The contrapositive of this result is that if S is not open under a structuring element A, then S is not open under the dilation of A with any other structuring element. Since $K_n \oplus J$ is the dilation of $(K_m \oplus J)$ with $(J_{m+1}$

$\oplus \cdots \oplus J_n$) we have the required result: if $S \neq (T_m \ominus J) \oplus J \oplus K_m$ then there can exist no decomposition of S involving J at this level m node or any node contained in the subtree below this level m node.

PROPOSITION 7. *$S \circ A = \{x \in S \mid$ for some t, $x \in A_t \subseteq S\}$.*

Proof. Suppose $x \in S$ and for some t, $x \in A_t \subseteq S$. Hence, for every $a \in A$, $a + t \in S$ and there exists a $y \in A$ such that $x = y + t$. But $a + t \in S$ for every $a \in A$ implies by definition of erosion that $t \in S \ominus A$. And $x = y + t$ with $y \in A$ and $t \in S \ominus A$ implies by definition of dilation that $x \in (S \ominus A) \oplus A$.

Suppose $x \in (S \ominus A) \oplus A$. Then there exists a $y \in S \ominus A$ and an $a \in A$ such that $x = y + a$. Since $y \in S \ominus A$, $y + z \in S$ for every $z \in A$, $A_y \subseteq S$. But $x = y + a$ with $a \in A$ implies $x \in A_y$.

PROPOSITION 8. *$S = S \circ (A \oplus B)$ implies $S = S \circ A$.*

Proof. Suppose $S = S \circ (A \oplus B)$. Let $x \in S$. By Proposition 7, there exists a t such that $x \in (A \oplus B)_t \subseteq S$. Hence there exists a $b \in B$ such that $x \in (A)_{t+b} \subseteq S$. By Proposition 7, $x \in S \circ A$. Thus, $S \subseteq S \circ A$. But by Proposition 4 it is true that $S \supseteq S \circ A$. Therefore, $S = S \circ A$.

III. 3. The Tree Search

In this section we give a complete description of the tree search using the understanding developed in Section III.2 plus a few additional computational efficiencies which follow immediately from what we already developed. When a level m node is born, it is given a name J_m, which is the mth structuring element in the decomposition of S. It is also given a heritage which consists of

(1) a restricted sequence L_m containing all of node m's future generation descendent name possibilities,

(2) the partial decomposition $K_m = J_1 \oplus \cdots \oplus J_m$, and

(3) the undecomposed part $T_m = S \ominus K_m$.

The sequence L_m is ordered according to the initial ordering $J^1, \ldots, J^M$ of all possible structuring elements that can participate in a decomposition, as discussed in Section III.2. The sequence L_m is, therefore, a subsequence of $\langle J^1, \ldots, J^M \rangle$. The partial decomposition satisfies $S \circ K_m = S$. Hence $T_m \oplus K_m = S$.

To accomplish the tree search, the node first goes through labor and then gives birth. The labor is accomplished by performing a forward check through the sequence L_m of all future generation children possibilities in the ancestral subtree below the node. The forward checking eliminates those children who cannot possibly participate in a decomposition solution. The reduced sequence is called L_m^* and it is generated by selecting each J in turn from L_m and then checking whether or not $S = (T_m \ominus J) \oplus (J \oplus K_m)$. The verification proceeds in three steps. It follows immediately from the definition of dilation that if $S = (T_m \ominus J) \oplus (J \oplus K_m)$, then it must necessarily follow that $\#S \leq \#(T_m \ominus J) \cdot \#(J \oplus K_m)$. So in the first step, $T_m \ominus J$ and $J \oplus K_m$ are computed and the inequality checked. If the inequality is not satisfied, J is not put into L_m^* and the next J from L_m is selected. If the inequality is satisfied, then the dilation $(T_m \ominus J) \oplus J$ is performed. If $(T_m \ominus J) \oplus J = T_m$, J is put into L_m^* for in this case we must have $[(T_m \ominus J) \oplus J]$

$\oplus K_m = T_m \oplus K_m = S$. If $(T_m \ominus J) \oplus J \neq T_m$, then the dilation $(T_m \ominus J) \oplus (J \oplus K_m)$ is performed and a comparison is made with S. If the comparison produces inequality J is not put into L_m^* and the next J is selected. If the comparison produces equality J is put into L_m^*.

After the forward checking labor is finished, the level m node is ready to give birth to its children. It sequences through the possible structuring elements in $L_m^* = \langle J^{i_1}, \ldots, J^{i_z} \rangle$ and gives birth to each of them in succession. Each birth gives rise to a level $m + 1$ node. The nth child is given the name J^{i_n}. That is $J_{m+1} = J^{i_n}$. For this child a new list L_{m+1} of future generation children is created. L_{m+1} consists of all the possibilities in L_m^* from the nth one to the last one. That is $L_{m+1} = \langle J^{i_n}, \ldots, J^{i_z} \rangle$. Then the partial decomposition K_{m+1} created by performing the computation $K_{m+1} = K_m \oplus J_{m+1}$. Then the reduced undecomposed part is created by performing the computation $T_{m+1} = T_m \ominus J_{m+1}$. The name J_{m+1} and the heritage L_{m+1}, K_{m+1}, and T_{m+1} are passed on to the level $m + 1$ node and the birth of the nth child is completed.

Actually, the computation of T_{m+1} associated with the nth child J^{i_n} is performed when it is verified that $S = (T_m \ominus J^{i_n}) \oplus (J^{i_n} \oplus K_m)$. But conceptually, the tree search description is easier to understand without thinking about that computational efficiency.

The tree search is done in a breadth-first manner and any decomposition found on the lowest level depth to produce a decomposition solution is an optimal decomposition. A decomposition solution is easily recognized since when it occurs on level N, $\#T_N = 1$ and the single member of T_N is the q appearing in the decomposition $S = \{q\} \oplus J_1 \oplus \cdots \oplus J_N$.

IV. TWO EXAMPLES

In the following we give two examples to diagrammatically explain the algorithm. The decomposition is done by two point sets.

EXAMPLE 1. Let S be the structuring element to be decomposed. Suppose S has the form $S = \{a, b, c, d\}$, where each element of S is a point in the plane and S represents the vertices of a parallelogram. See Fig. 1.

It is clear that L_0 (root) contains four two point sets, J^1, J^2, J^3, J^4, where

$$J^1 = \{0, b - a\}$$
$$J^2 = \{0, c - b\}$$
$$J^3 = \{0, c - a\}$$
$$J^4 = \{0, d - b\}$$

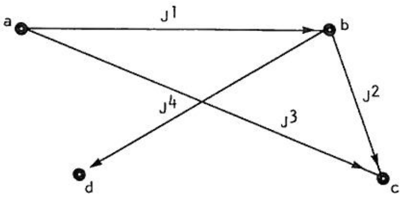

FIG. 1. Illustrates the geometry of the structuring element S for Example 1.

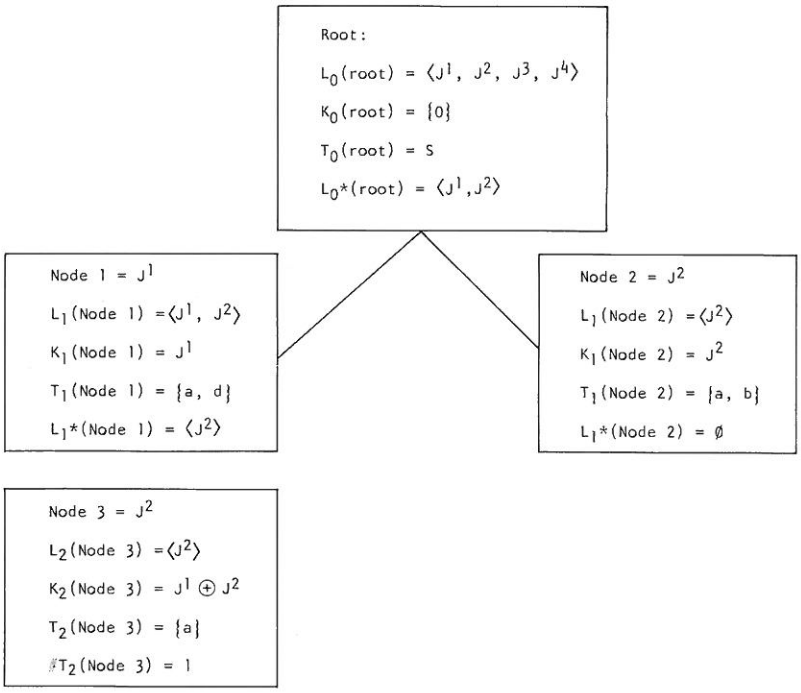
Root:
$L_0(\text{root}) = \langle J^1, J^2, J^3, J^4\rangle$
$K_0(\text{root}) = \{0\}$
$T_0(\text{root}) = S$
$L_0{*}(\text{root}) = \langle J^1, J^2\rangle$
Node 1 = J^1
$L_1(\text{Node 1}) = \langle J^1, J^2\rangle$
$K_1(\text{Node 1}) = J^1$
$T_1(\text{Node 1}) = \{a, d\}$
$L_1{*}(\text{Node 1}) = \langle J^2\rangle$
Node 2 = J^2
$L_1(\text{Node 2}) = \langle J^2\rangle$
$K_1(\text{Node 2}) = J^2$
$T_1(\text{Node 2}) = \{a, b\}$
$L_1{*}(\text{Node 2}) = \emptyset$
Node 3 = J^2
$L_2(\text{Node 3}) = \langle J^2\rangle$
$K_2(\text{Node 3}) = J^1 \oplus J^2$
$T_2(\text{Node 3}) = \{a\}$
$\#T_2(\text{Node 3}) = 1$

FIG. 2. Illustrates the tree search for the S in example 1.

since neglecting signs, $b - a$, $c - b$, $c - a$, and $d - b$ constitute all the differences between pairs of elements of S. It is not necessary to consider the following point sets: $\{0, a - b\}$, $\{0, b - c\}$, $\{0, a - c\}$, $\{0, b - d\}$, since they are all translates of J^1, J^2, J^3, J^4. See Fig. 2.

The optimal decomposition of S: $S = \{a\} \oplus J^1 \oplus J^2$.

EXAMPLE 2. Let S be a 5×3 rectangle whose corner points are missing. Suppose S has a form $S = \{a_1, a_2, a_3, b_1, b_2, b_3, b_4, b_5, c_1, c_2, c_3\}$, where the a's represent equally spaced points on the top row, the b's represent equally spaced points on the middle row, and the c's represent equally spaced points on the bottom

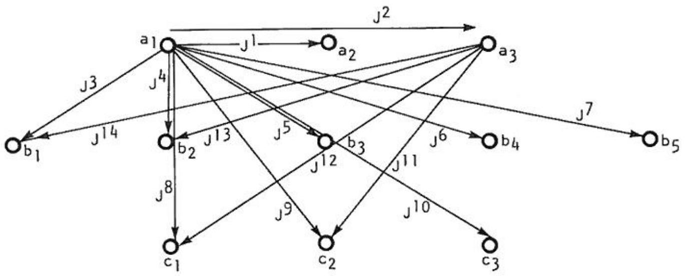

FIG. 3. Illustrates the geometry of the structuring element S of Example 2.

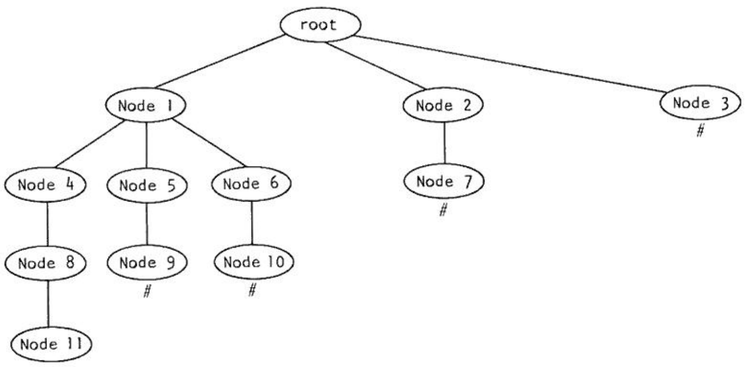

FIG. 4. Illustrates the tree search for Example 2. The # sign indicates the node dies.

root:
$L_0(\text{root}) = \langle J^i : 1 \leq i \leq 14 \rangle$
$K_0(\text{root}) = \{0\}$
$T_0(\text{root}) = S$
$L_0^*(\text{root}) = \langle J^1, J^3, J^5 \rangle$

Node 1 $= J_1$
$L_1(\text{Node 1}) = \langle J^1, J^3, J^5 \rangle$
$K_1(\text{Node 1}) = J^1$
$T_1(\text{Node 1}) = \{a_1, a_2, b_1, b_2, b_3, b_4, c_1, c_2\}$
$L_1^*(\text{Node 1}) = \langle J^1, J^3, J^5 \rangle$

Node 2 $= J^3$
$L_1(\text{Node 2}) = \langle J^3, J^5 \rangle$
$K_1(\text{Node 2}) = J^3$
$T_1(\text{Node 2}) = \{a_1, a_2, a_3, b_3, b_4, b_5\}$
$L_1^*(\text{Node 2}) = \langle J^5 \rangle$

Node 3 $= J^5$
$L_1(\text{Node 3}) = \langle J^5 \rangle$
$K_1(\text{Node 3}) = J^5$
$T_1(\text{Node 3}) = \{a_1, a_2, a_3, b_1, b_2, b_3\}$
$L_1^*(\text{Node 3}) = \varnothing$

Node 4 $= J^1$
$L_2(\text{Node 4}) = \langle J^1, J^3, J^5 \rangle$
$K_2(\text{Node 4}) = J^1 \oplus J^1$
$T_2(\text{Node 4}) = \{a_1, b_1, b_2, b_3, c_1\}$
$L_2^*(\text{Node 4}) = \langle J^3, J^5 \rangle$

Node 5 $= J^3$
$L_2(\text{Node 5}) = \langle J^3, J^5 \rangle$
$K_2(\text{Node 5}) = J^1 \oplus J^3$
$T_2(\text{Node 5}) = \{a_1, a_2, b_3, b_4\}$
$L_2^*(\text{Node 5}) = \langle J^5 \rangle$

Node 6 $= J^5$
$L_2(\text{Node 6}) = \langle J^5 \rangle$
$K_2(\text{Node 6}) = J^1 \oplus J^5$
$T_2(\text{Node 6}) = \{a_1, a_2, b_1, b_2\}$
$L_2^*(\text{Node 6}) = \varnothing$

Node 7 $= J^5$
$L_2(\text{Node 7}) = \langle J^5 \rangle$
$K_2(\text{Node 7}) = J^3 \oplus J^5$
$T_2(\text{Node 7}) = \{a_1, a_2, a_3\}$
$L_2^*(\text{Node 7}) = \varnothing$

Node 8 $= J^3$
$L_1(\text{Node 8}) = \langle J^3, J^5 \rangle$
$K_3(\text{Node 8}) = J^1 \oplus J^1 \oplus J^3$
$T_3(\text{Node 8}) = \{a_1, b_3\}$
$L_3^*(\text{Node 8}) = \langle J^5 \rangle$

Node 9 $= J^5$
$L_3(\text{Node 9}) = \langle J^5 \rangle$
$K_3(\text{Node 9}) = J^1 \oplus J^1 \oplus J^5$
$T_3(\text{Node 9}) = \{a_1, b_1\}$
$L_3^*(\text{Node 9}) = \varnothing$

Node 10 $= J^5$
$L_3(\text{Node 10}) = \langle J^5 \rangle$
$K_3(\text{Node 10}) = J^1 \oplus J^3 \oplus J^5$
$T_3(\text{Node 10}) = \{a_1, a_2\}$

Node 11 $= J^5$
$L_4(\text{Node 11}) = \langle J^5 \rangle$
$K_4(\text{Node 11}) = J^1 \oplus J^1 \oplus J^3 \oplus J^5$
$T_4(\text{Node 11}) = \{a_1\}$
$\#T_4(\text{Node 11}) = 1$

FIG. 5. Illustrates the state of the search for each node in the search tree of Example 2.

row. See Fig. 3. It is easy to see that L_0 (root) $= \langle J^i: 1 \le i \le 14 \rangle$, where

$$\begin{array}{lll} J^1 = \{0, a_2 - a_1\}, & J^2 = \{0, a_3 - a_1\}, & J^3 = \{0, b_1 - a_1\}, \\ J^4 = \{0, b_2 - a_1\}, & J^5 = \{0, b_3 - a_1\}, & J^6 = \{0, b_4 - a_1\}, \\ J^7 = \{0, b_5 - a_1\}, & J^8 = \{0, c_1 - a_1\}, & J^9 = \{0, c_2 - a_1\}, \\ J^{10} = \{0, c_3 - a_1\}, & J^{11} = \{0, c_2 - a_3\}, & J^{12} = \{0, c_1 - a_3\}, \\ J^{13} = \{0, b_2 - a_3\}, & J^{14} = \{0, b_1 - a_3\}. & \end{array}$$

See also Figs. 4 and 5.

The optimal decomposition of S: $S = \{a_1\} \oplus J^1 \oplus J^1 \oplus J^3 \oplus J^5$.

V. CONCLUSION

We have given a complete and optimal solution to the morphological decomposition of a structuring element by dilations. This decomposition problem has the same value for hardware performing erosions and dilations as the iterated kernel decomposition problem has for hardware performing convolutions. In the convolution case, a large kernel is given and it must be performed on hardware which can only perform small kernel convolutions quickly. In the morphology case, a large structuring element is given and a morphological erosion or dilation must be performed with it on hardware which is only capable of executing small structuring element morphological operations quickly.

The essence of the solution technique was (1) the recognition that structuring elements participating in the decomposition must have members which are the differences between members of the given structuring element and (2) that it is necessary for the undecomposed part of the structuring element to be morphologically open with respect to any structuring element participating in its further decomposition. Based on these two facts plus some basic properties of morphological operations, it was possible to describe a reasonably efficient tree search to determine the optimal decompositions.

REFERENCES

1. J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, New York, 1982.
2. R. M. Loughheed, D. L. McCubbrey, and S. R. Sternberg, Cytocomputers: Architectures for parallel image processing, in *Proceedings*, *Workshop Picture Data Descr. and Management*, Pacific Grove, Calif. Aug. 27–28, 1980, pp. 282–286.
3. S. Sternberg, Cellular computers and biomedical image processing, in *Biomedical Images and Computers. Proceedings*, *1980* (J. Sklansky and J. C. Bisonte, Eds.), Lecture Notes in Medical Informatics Vol. 17, pp. 274–319, Springer-Verlag, Berlin, 1980.
4. S. R. Sternberg, Pipeline architectures for image processing, in *Multicomputers and Image Processing-Algorithms and Programs* (L. Uhr, Ed.), pp. 291–305, Academic Press, New York, 1982.
5. S. R. Sternberg, Languages and architectures for parallel image processing, in *Proceedings*, *Conference on Pattern Recognition in Practice*, Amsterdam, May 21–23, 1980 (L. N. Kanal and E. S. Gelsema, Eds.), North-Holland, Amsterdam, 1980.
6. R. M. Haralick and G. L. Elliott, Increasing tree search efficiency for constraint satisfaction problems, *Artif. Intell.* **14**, 1980, 263–313.