製作施用於階層式形態影像處理之對照表
Building a Look-up Table for Hierarchical Morphological Image Processing

周本生
Chow Ben-Shung
Mail:bschow@ee.nsysu.edu.tw

鍾明峰
Chung Ming_Fung
Phone: (07)5252000-4172

黃聖賢
Huang Sheng_xian
Phone: (07)5252000-4172

中山大學電機系
National Sun Yat-sen UniversityDepartment of Electrical Engineering
Kaohsiung, Taiwan, ROC

摘要

爲了在四分樹上做擴張運算，將會遇到原影像中的黑葉因遭遇擴張運算中的位移，而使得原來的黑葉變成複雜的子樹，我們解決這個問題的方式是預先就準備好這一類的樹圖案，基本上這是一種「查表」的觀念，把所有可能發生的情形分析歸納，整理儲存在一個資料庫內。等到線上處理時就可以直接由樹圖案資料庫取出樹圖案再掛上工作樹，這種運算的方式非常地有效率，而形態影像處理的速度也將因四分樹的使用會有顯著的改善。而基於實用的資料庫其資料量愈簡潔，使用上也愈經濟愈好的考量下，我們也針對資料庫作了細心的評估、選擇與設計，其中利用擴張運算的交換性，以及原影像與結構元素黑葉間的比例關係，加上運用二分樹替代四分樹的方案，有效縮減了四分樹圖案的體積，最後達到約 20Kbytes 的量。在線上處理時間的評估方面，我們與另外兩種演算法作實際的電腦模擬實驗比較，由明確的數據，分析我們演算法的優劣所在。

關鍵詞：形態影像處理，階層式資料架構，查表法.

Abstract

The picture in a quadtree data structure is stored in the form of blocks because every leaf in the tree stands for a square block. Thus, dilation of the whole image can be accomplished by manyfolds dilating individual decomposed square blocks. However, there is alwayse a translation vector associated to each fold of the dilating process according to the property of translation invariance. This translation causes a complexity for the dilated results of the subtrees. These subtrees are proposed to be prepared in a look-up table for online use. In applications, nothing needs to be done for the white square blocks, but a simple dilation by look-uping the table is required for black square blocks. To reduce the size of look-up table, pattern reduction is investigated. Some new observations based upon tree structure and geometry are developed in this paper to validate our pattern reduction strategy. It is noted that this single table works for all different pictures.As a summary, it is the quadtree data structure to validate our blocks vs. blocks type of morphlogical image processing and it is our look-up table method to make this processing under quadtree data structure even faster. The proposed quadtree method will be compared with the direct method and the pyramid approach.

Keywords: Morphological image processing, Hierarchical data structure, Look-up table

I. Introduction

The morphological image processing[1] is a type of processing by which the spatial forms or structures of objects within an image are modified. The way they are modified are usually nonlinear. The computation cost of nonlinear processing for conventional numerical image processing is very expensive. However, the morphology image processing treats the image components as sets and deals with the change of shape very efficiently. The quadtree data[2] structure is based on the principle of recursive decomposition's of spatial data. In this paper, the quadtree data structure will be further applied to the implementation of morphological image processing since an efficient data structure is important for morphological processing.

A picture in its quadtree data structure is stored in a form of blocks since every leaf in the tree stands for a square block. Owing to the law of distributivity of dilation over union, the morphological image processing can be processed on the basis of a pair of squares' dilation (square vs. square dilation) for the case of the input image and the structuring element both consisting of the quadtrtee data structure. Thus, dilation of the whole image can be accomplished by manyfolds dilating individual decomposed square blocks. However, there is alwayse a translation vector associated to each fold of the dilating process according to the property of translation invariance. This translation vector is due to the location vector of the corresponding decomposed block in the structuring element. This translation causes a complexity for the dilated results of the subtrees. These subtrees are proposed to be prepared in a look-up table for online use. In applications, nothing needs to be done for the white square blocks; a simple dilation is performed by look-uping the above table for black square blocks.

To reduce the size of look-up table, pattern reduction is investigated intensetively. Some well-known dilation properties such as translation invariance and communitivity of dilation are applied. Most importantly, some new observations based upon tree structure and geometry are developed in this paper to validate our pattern reduction strategy. They are the concepts of formatted region, tree identity from pattern similarity , and spliterting the quadtree string into two binary-tree strings.

Pyramid[3] is conceptually a very similar data structure compared to the quadtree data structure. Pyramid structure is a sequence of arrays, in which the image is presented in different resolutions, and thus pays more price for spatial storage than the quadtree since the pyramid is equivalent to a complete quadtree [4]. For example, a black node for the quadtree need to extend to four black sons in pyramid structure. However, this spatial inefficiency results in the pyramid's application of the starting point for a 'split-and-merge' segmentation algorithm. In contrast, the string quadtree is very spatially efficient and is an important compression standard for binary images. We also implement another hirerarchical processing by a pyramid approach. Research for the combination of the quadtree and morphology has not been found in the literature. However, our pyramid approach is very much related to the original work of Liang and Wang [5].

The proposed quadtree approach will be compared with the direct approach [6] and the pyramid approach. The experiments are not just set up for comparing the speed of three methods. Furthermore, the effects from the characteristics of the input images and structuring elements are investigated for three methods. In this sense, the experiment is set up for the characteristic analysis for the three approaches. Thus, it should be noted that the three approaches should not be evaluated by the speed reports. Actually, each of the three approaches has its own appropriate situation and its irreplaceable standing for its applications. The direct approach has its advantage of simple implementation and general applications in windows environments. The pyramid approach has its privilege in the applications of progressive transmission and presentation. The quadtree approach is suitable for progressive transmission and data compression since the string quadtree data structure is a widely accepted standard for binary data compression. Thus, in the proposed quadtree approach, the input image, output image, and structuring element are all arranged in the string quadtree data structure.

II. Square as Basics for Dilation on the Quadtree

Regular morphological image processing is processed on a pixel by pixel basis. This is due to the fact that the picture is stored in an array form and thus the pixel is accessed by pixel. In this sense, if the picture is stored in a form of blocks, the morphological image processing can be processed on the block by block basis. In the quadtree data structure, the morphological image processing can be processed on the basis of the pairs of squares (square vs. square dilation) if the quadtree data structure is used in the image and the structuring element. It should be remembered that the above mentioned dilation of squares involves the translation operation and needs more investigation.

Difficulty from the Picture Translation in the Quadtree.

Translation for the picture in array is conceptually simple but mechanically inefficient since it is performed pixel by pixel. For every pixel, the processing of location calculation, data retrieving, and data storeing need to be carried out. In contrast, the translation for quadtree case is conceptually difficult but mechanically simple or complicated dependending upon the the translation vector. This fact needs more investigation.

Morton order is so much different from the raser scan order. Therefore, it can be observed that spatially neighbouring pixels are not necessarily adjecent in Morton order. This observation implies that a block of pixels, depending upon its location, may be or may be not a bundle of adjecent leaves, which can be merged as one leaf. Thus the following idea of formatted region is introduced.

Formatted Region

The blocks corresponding to the leaves in the quadtree are the results from the recursive decomposition of the original picture. The procedure of recursive decomposition determines the idea of formatted region, which is the fixed location where the leaf blocks may reside. For example, a block of the size $2^n * 2^n$ is only allowed to reside in the formatted region with upper left conner located in coordinates (x,y) where x and y are multiples of 2^n. The idea of formatted region causes the work of translation of a block to be classified in two cases: formatted translation and nonformatted translation.

Look-up Table Method

A pattern mapping method is introduced in this paper to solve the above translation problem for facilitating the on-line square vs. square processing. The result of an image square dilated by a structuring square is defined as a pattern, which can be off-line computed in advance and stored in the look-up table for on-line application.

III. Table Reduction and Lookingup

Table reduction is important for the pratical reason since it determines the size of look-up table. Some well-known dilation properties are exploited. Most importantly, some new observations are developed in this section to validate our pattern reduction strategy. For clearance, the reduction procedures are introduced step by step as follows.

Distributivity of Dilation over Summation

$$A \oplus B = \bigcup_i \bigcup_j (Ai \oplus Bj)$$

By this property, the whole processing can bedecomposed into many individual processings. Each of them is a dilation of the image leaf Ai and the structuring element leaf Bj. The above decomposed dilation results can be found in the look-up table where all the quadtree patterns of the standard dilated squares are stored for on-line applications.

The reduction analysis is thus sarted from the formulation of $Ai \oplus Bj$. It is clear that there are 9 different sizes and 512*512 kinds of translations for Ai and Bj respectively for the case of the picture being assumed to be of the size 512*512. A very rough beginning of the counting for the number of patterns is thus

$$9*9*512*512*512*512.$$

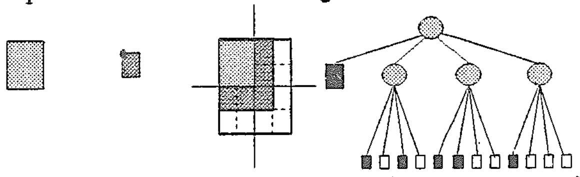
Translation Invariance

$$(Ai+t1) \oplus (Bj+t2) = (Ai \oplus Bj) + (t1+t2)$$

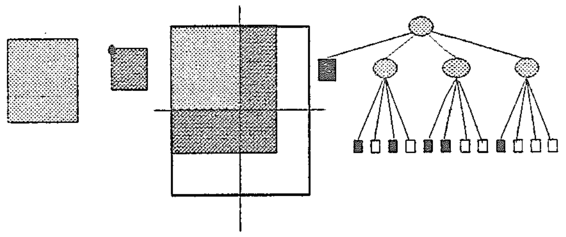
The translation invariance tells us the relation between the results of translation before and after the dilation. By this property two translation vectors can be reduced to one vector. Hence, the number of patterns reduces to 9*9*512*512.

Tree Identity for Pattern Similarity from Inputs Propertionality

It is observed that the dilated results for two pairs of blocks will have the same quadtree representation if the ratio of the sizes for the image and the structuring block are the same between each pair. For example, two input pairs (A1,B1) and (A2,B2) in Figure 1, with their sizes being different but size ratio being the same. The two dilated results are similar with each other and have a same quadtree representation as shown in Figure 1.



(a) A1 (b) B2 (c) $A1 \oplus B1$ (d)quad-tree representation for $A1 \oplus B1$



(a')A2 (b')B2 (c') $A2 \oplus B2$ (d')quad-tree representation of $A2 \oplus B2$

Figure 1. An example of illustrating the tree identity due to the pattern similarity.

Since the dilated results is determined by the size ratio of Ai and Bj, patterns can be considered only for the pair with one block fixed. Thus, two variables are reduced to one. In this sence, the number of patterns becomes

$$2*9*512*512$$

where 2*9 represents the number of possible kinds of ratio between Ai and Bj.

Communtivity of Dilation

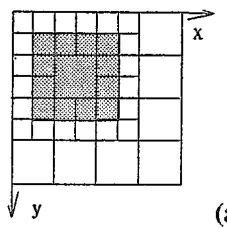
$$(Ai \oplus Bj) = (Bj \oplus Ai)$$

By this property, the roles of the image and the structuring element can be switched. Hence, the factor 2 can be dropped and thus make the number of patterns to be

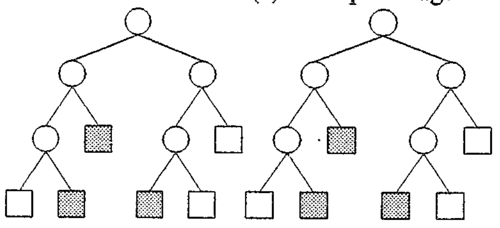
$$9*512*512$$

Splitting Quadtree into Two Binary Trees

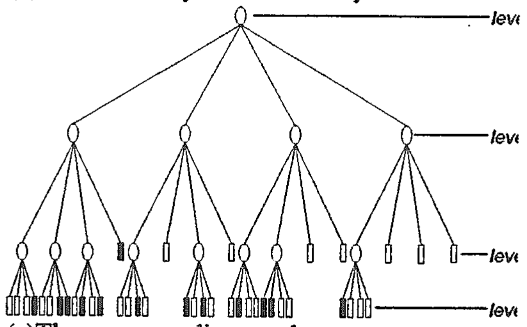
It can be further observed that the dilated results are always square shaped. A two dimention square function is separable. As a result, the dilated results are capable of being expressed as the product of x and y distributation.



(a)A sample image



(b) a same binary tree for x and y dimension



(c)The corresponding quadtree

Figure 2. An example showing the quadtree being splitted to two binary tree.

Consequently, the large quadtree pattern is separable and reduced to two small binary trees. Thus, the number of patterns are reduced to

9*512*2

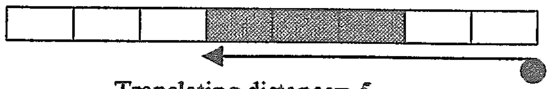
since a two-dimention vector of 512*512 kinds is reduced to two one-dimention vectors of 512 kinds. After a simple thought, the factor two can be droped because x and y dimetion can share the same binary patterns. Thus, the number of patterns is

9*512

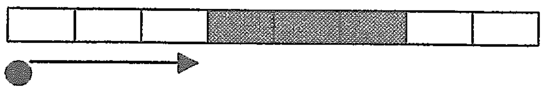
Geometry Consideration

Consider the complementary relation exiting in the translation vectors shown below. The number of patterns becomes

9*256



Translating distance=-5



Translating distance=3

Figure 3. An example for geometry equivalence.

Possible Translations

Now, the physical meanings for 9*256 are reviewed for further reduction. The value 9 is the number of kinds of block ratios. Accordingly, 256 is the number of different translations. After a liltle thought, the possible translations are different for different block ratios. To be consise, the number of possible translation for a specific ratio is exactly the value of ratio itself. A table summarizing the possible translations is shown as below.

Class	Image block	S.E. Block	Possible tran.	#of N
1	2^8	2^8	0	1
2	2^8	2^7	$2^7 \times n$	2
3	2^8	2^6	$2^6 \times n$	4
4	2^8	2^5	$2^5 \times n$	8
5	2^8	2^4	$2^4 \times n$	16
6	2^8	2^3	$2^3 \times n$	32
7	2^8	2^2	$2^2 \times n$	64
8	2^8	2^1	$2^1 \times n$	128
9	2^8	2^0	$2^0 \times n$	256

Table 1. Classification of patterns for dilated results

Therefore, instead of using the multiplication in 9*256 , we accomlish the following summation to count the number of patterns

1 + 2 + 4 + 8 +16 +32 +32 + 64 +128 + 256

that is

(1/256 + 2/256 + ... + 256/256) = 2*256.

It should be noted that, not only the number of patterns is reduced from $9*9*(512*512)^2$ to 2*256, but also the patterns themselves are shrinked from quadtrees to binary trees. The size of a binary tree is much smaller than that of the quadtree.

Table Looking-up

Recalling the description in the section of pattern reduction, the larger block is alwayse normalized to the block with the size of level one. Thus, the idea of level difference is introduced to magnify the smaller block. Afterwords, there is a vector of virtual translation associated to this magnified block and the corresponding pattern is accordingly seclected.

Therefore, the look-up table for pattern selection has two index. The first index is a scalar d, the level difference. The sececond index is a vector (x,y), the vector of virtual translation with x and y as

$x = (X*2^d) \%256$

$y = (Y*2^d)\%256$

where X an Y are the coordinates of the upper left conner of the hanning block, i.e.,

$X = xA + xB$

$Y = yA + yB,$

where xA, yA, xB, and yB are x and y coordinates of the blocks A and B.

IV. Experiments and Comparisons

Experiments Setup

All these experiments are tested on two images: a natural image, the image key , and a generated image, the image segments, in Figure 7.1.

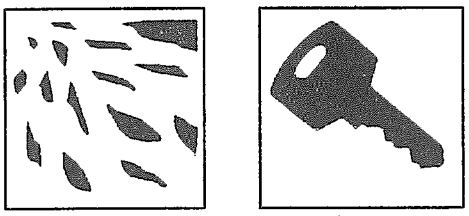

(a) (b)

Figure 4. Testing input images (a) the image fragment, and (b) the image key.

Our method is further compared with two other morphological processing methods from the above perspectives mentioned above. The first comparing basis is a direct method [6], which is processed by a pixel vs. pixel basis, built directly on the definition. The second but most important basis is a pyramid method, which is equivalently based upon the block vs. pixel basis.

The execution time for our method is counted from the stage of picture being stored in the string form to the stage of string again. For pyramid approach and direct method the time are counted from the time nine hierarchical arrays has been prepared and one array picture has been ready respectively. In other words, three methods are compared on the grounds of their different favorable data structures.

All the experiments are C codes implemented in the platform of Pentium 133 machine with operating system Linux 3.2.

Experiments Results and Analysis

The square structuring elements are selected for our condensed type structuring elements. There are five sizes of square structuring elements chosen starting from

$$2^0 * 2^0 \text{ to } 2^4 * 2^4.$$

The execution time of the image key and the image fragments for three methods are listed in the table 2.

Size of S.E.	Proposed approach	Pyramid approach	Direct approach
1×1	0.20	0.18	0.18
2×2	0.18	0.19	0.45
4×4	0.19	0.33	1.41
8×8	0.23	0.16	4.91
16×16	0.30	7.19	18.52

(a)

Size of S.E.	Proposed approach	Pyramid approach	Direct approach
1×1	0.16	0.16	0.18
2×2	0.14	0.16	0.46
4×4	0.13	0.22	1.41
8×8	0.15	0.57	4.91
16×16	0.18	2.82	18.53

(b)

Table 2. (a) Computation time in sec. of three methods for the image fragments.(b) Computation time in sec. of three methods for the image key.

It is shown that the proposed approach is slower than the pyramid approach only for very small structuring elements' cases (with the size less than 2*2) but will be much faster for large elements cases. direct methods is the slowest for all cases.

The reason for direct method's slowness can be easily understood by its pixel vs. pisel's basis. But, explanations for the other two methods are not straight forward. It is shown that the size of structuring elements give significant effect on pyramid approach. This is because on the lowest level all the pixels of the structuring element are expressed out and is processed with the undetermined pixels of the image. Also, the lowest level processing is the bottle neck of pyramid approach. Since the size of the structuring element is exactly the number of the pixels of the structuring element, the larger the size of the element, the longer time it takes for pyramid approach. For our method, the size of the element in our experiment does not change the number of nodes and thus does not effect the execution time greatly. However, a big size of structuring element paired with a small image block, for instance 1*1, will correspond to a more complicated quadtree pattern and thus explains the phenomenon that our execution time increase with the size of the structuring element but not as much as the other two methods.

The image key needs more time for processing than the image fragments for both our method and pyramid approach. This is because the image key is a condensed type of image and thus is advantages for hierarchical presentation. If the above mentioned effect of structuring is further considered for pyramid approach, It is found that the effect is more significant in the fragments image than the key image. One more reason other than the hierarchical reason needs to be given, white leaves dominates the computation in pyramid approach and it is observed that the image fragments are more white dominate than the key image.

V. Conclusions

The proposed method is the fastest one among the three comparing methods except for the very small structuring elements case, as demonstrated in the experiments results. It is the total number of pixels in the structuring element that determines the execution time for the former two methods but it is the number of black nodes in the structuring element for our method. The proposed method is therefore prevailing in light of the fact that the number of nodes is usually much less than the number of pixels. For example, the number of pixels in a single node of the n'th level from the bottom is as many as n square.

The efficiency of look-up table method is a tradeoff between the table size and the applying speed. By a successful strategy of pattern reduction the size of table is around 20k bytes for 512*512 image handling capability. This 20 k table is not image dependent and is one for all images. Furthermore, the 20k size is relatively small compared to the original image size about 200k for the baseline storage's of the direct and the pyramid methods. On the other hand, the applying speed is efficient by our pattern developing method implemented on the pointer working tree. As a summary, the pattern is stored in x and y binary tree forms of string structure to achieve pattern reduction. For processing efficiency the working tree is implemented in the pointer quadtree data structure. Also, for data compression, the input image, output image, and structuring element are all arranged in the string quadtree data structure.

XI. References

[1] Serra, J. "Image Analysis and Mathematical Morphology", New York: Academic Press, 1983

[2] Samet, "Application of Spatial Sata Structure Computer Graphic, Image Processing, and GIS", Addison Wesley 1990

[3] S. Tanimoto and T. Pavlidis, "A Hierarchical Data Structure for Picture Processing, Computer Graphics and Image Processing 4", June 1975, pp. 104-119.

[4] M. Pietikainen, A. Rosenfeld, and I.Water, split- and -link algorithms for image segmentation, Pattern Recognition 15,4, 1982, pp. 287- 298.

[4] En-Hui Liang and Edward K. Wong "Hierarchical algorithms for morphological logical image processing. "Pattern Recognition, 1993, vol. 26, no. 4, pp.511-529.

[6] R. Myler and A. R. Weeks, Computer Imagint Recipes in C, Prentice-Hall Inc. NJ, 1993.