International Journal of Modern Engineering Research (IJMER)
www.ijmer.com Vol.2, Issue.2, Mar-Apr 2012 pp-354-357 ISSN: 2249-6645

An Analysis of Morphological Operation Using Arbitrary Structuring Element

A.P. Ilayaraja, M. P. Gopinath
School of computing science and Engineering, VIT University, India.

ABSTRACT
Mathematical morphology with spatially variant structuring elements outperforms translation-invariant structuring elements in various applications. However, supporting a variable structuring element shape imposes an overwhelming computational complexity, dramatically increasing with the size of the structuring element. Dilation and erosion are often used in combination to implement image processing operations. The image content and the number of gray levels used does not influence the computing time required. The operator for each size and shape of the structuring element must be done separately, but here the filtering with multiple structuring elements is done in one operation and this reduces the computational requirements very much. The method finds applications in the areas like granulometrices, dilation-erosion scale spaces, and template matching using the hit-or-miss to transform. In this paper we are proposing about the computational performance advantage over existing methods where structuring elements are used that cannot be easily decomposed into linear structuring elements.

Keywords **- Dilations, Erosions, Granulometrices, Hit-Or-Miss Transforms, Mathematical Morphology, Structuring Elements.**

I. INTRODUCTION
Mathematical Morphological have a most fundamental morphological operators [9] like dilation and erosion with structuring elements (S.E.). It has become common tools for both image filtering and analysis of binary and gray-scale images since the development of efficient algorithms [2]. It is based on the algebra of nonlinear operators operating on objects shape and in many respects to take place the liner algebra system of convolution. Morphological operations simplify images, and quantify and preserve the main shape characteristics of objects. Morphological operations are applied to image pre-processing, enhancing the object structure, segmenting objects from the background, quantitative description of objects. Algorithms are fixed to liner structuring element and shapes like rectangles that can be decomposed [4] in to a series of linear structuring elements. All methods based on decomposition of 2-D S.E.s [3] into linear S.E.s share the same limitation: many shapes they cannot be decomposed at all or either cannot be decomposed efficiently.

One of the very interesting and effective algorithms for the speed calculation on established computers of the basic morphological operation for 2-D images is presented in [5]. Still, that algorithm cannot be extended directly to 3-D images, because chain coding has not an equivalent in three dimensions. In the binary case, efficient algorithms for some 2-D shapes like circles do exist, but these cannot efficiently be extended to the gray-scale case, for which polygonal approximations of circles usually are used instead. Because larger circles this estimation is given to be either too coarse or too computationally intensive, since the number or liner S.E.s required is proportional to the diameter of the circle. Van Droogenbroeck and Talbot [1] proposed an efficient algorithm for computing morphological operation with arbitrary 2-D shapes using a histogram, which makes the computing time of their algorithm dependent on the number of gray levels used. Efficient implementations for particularized hardware have also been analysed extensively, such as the decomposition of arbitrary shapes into $\mathbf{3X3}$ blocks [6]. The important for those case where S.E. cannot be decomposed, where the algorithms that efficiently perform morphological operations with arbitrary S.E.s. but also where ever a generic algorithm is desired such as image processing libraries, which often have number of specialized routines for specific cases, and a direct implementation for arbitrary S.E. moreover, for many application the benefits of using the faster specialized algorithm available alternatively of using one a bit slightly less efficient generic algorithm does not outbalance the coast involved in adapting the methods used. S.E. shape decompositions require some design and programming efforts that can be avoided if a generic algorithm is used [7]. To calculate for one pixel p of the image the complete histogram based on the intensity of the pixels around p matching the element of the S.E. After erosion the value of p is the minimum intensity in the histogram which has a value >0. For all succeeding pixels of the image, the histogram is efficiently update and the position of its minimum intensity changes only if i) a new minimum value is shifted into the histogram, which can be kept track when the histogram is updated, or ii) when the current minimum is shifted out of the histogram, in which case the algorithm searches for the first following intensity which is now represented in the histogram. This paper [8], present a new technique for performing morphological operators with any 2-D structuring element that always outperforms existing algorithms for arbitrary structuring elements, and which has two advantages: i) it is independent of both image content and the number of gray levels used, and ii) application of a single operator using many different S.E.s can be computed somewhat more efficiently, which may be useful for granulometries [9, 2] and erosion dilation scale spaces [10]. Compared to Van Droogenbroeck and Talbot's method, it has the further advantage that it also works on floating point data.

II. ALGORITHM

Discussion here to discrete 2-D gray scale images $\boldsymbol{f}$ and erosion with 2-D flat S.E.s $\boldsymbol{A}$. Be noted that our method can be easily modified for 3-D images and other morphological operation. All images $\boldsymbol{f}$ have their origin top-left and that images are processed in scan-line order.

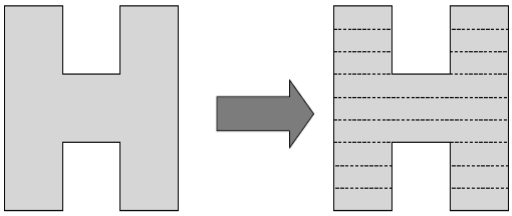


Fig. 1. decomposition of S.E. A into chords

Like the previous method, our approach to enhance the computational efficiency of erosion is by reducing the number the repeated comparisons performed by the implementation of its definition.

$$(f \ominus A) = \min_{z \in A} f\,(p + z) \qquad (1)$$

Our algorithm disintegrates an arbitrary S.E. into a series of chords, i.e., runs of highlight pixels of maximum level as demonstrated in fig. for letter H. each chord is represented by a triplet containing.(i) its $\boldsymbol{y}$-offset with respect to the origin of the S.E.,(ii) its minimal $\boldsymbol{x}$-position, and (iii) its length $\boldsymbol{l}$. For each S.E. the number chords the minimum and maximum $\boldsymbol{y}$-offsets $\boldsymbol{y_{min}}$ and $\boldsymbol{y_{max}}$, and minimum and maximum and minimum and maximum $\boldsymbol{x}$-values $\boldsymbol{x_{min}}$ and $\boldsymbol{x_{max}}$,and the maximum chord length $\boldsymbol{l_{max}}$ occurring in$\boldsymbol{A}$. Obviously, for each shifted S.E. we can compute the minimum value within the each chord, and compute for all chords with minimum of all chord-minimum, and also we compared minimum of all pixels within the shifted S.E. Other way to do this when a image processing in row y, is by creating an auxiliary 2-D array $\boldsymbol{W_v(i,x)}$ for each image row $\boldsymbol{v}$ between $\boldsymbol{y - y_{min}}$ and $\boldsymbol{y - y_{max}}$. Each array start at index $\boldsymbol{i}$ runs from 0 to $\boldsymbol{l_{max}}$, inclusive, and $\boldsymbol{x}$ from $\boldsymbol{x_{min}}$ to $\boldsymbol{X + x_{max} - 1}$, including , with $\boldsymbol{X}$ the $\boldsymbol{X}$ dimension of image $\boldsymbol{f}$. Each value $\boldsymbol{W_v(i,x)}$ is defined asy

$$\boldsymbol{W_v(i,x) = \min_{u \in [x, x+i]} f(u,v)} \qquad (2)$$

In each array cab be computed in $\boldsymbol{l_{max}}$ comparisons per pixel.now compute the minimum over in $\boldsymbol{N_c - 1}$ further comparisons per pixel. We compute for each value of $\boldsymbol{x}$ is

$$\boldsymbol{(f \ominus A)(x,y) = \min_{j \in \{0,1,\ldots N_c - 1\}} W_{y+y_j}(l_j, x + x_{min,j})} \qquad (3)$$

In which$\boldsymbol{y_j}$, $\boldsymbol{x_{min,j}}$ and $\boldsymbol{l_j}$ denote the $\boldsymbol{y}$-offset, length $\boldsymbol{l}$ of chord $\boldsymbol{j}$ and minimum $\boldsymbol{x}$ position. This version has a computational complexity per pixel of $\boldsymbol{O(l_{max} + N_c)}$.

```
/* Copy image data into    Wv(0, :)*/
for (x = 0; x < X; x++)
Wv(0, x) = f(x,v );
/* Pad copied data on either side of copied row */
for (x = xmin; x < 0; x++)
Wv(0, x) = f(0,v);
for (x = X; x < X + xmax − 1; x++)
Wv(0, x) = f(X − 1, v);
/* Compute minima of runs of length 2n, starting at each x
*/
for (i = 1; i ≤ _log2(lmax − 1)_; i++)
for (x = xmin; x < X + xmax − 1 − 2i−1; x++)
Wv(i, x) = min(Wv (i−1, x),Wv (i−1, x+2i−1))
```

Fig. 2. Pseudo code for computing $\boldsymbol{W_v}$ in time proportional to $\boldsymbol{log(l_{max})}$

We can reduce both complexities using the following observation. We can compute the minimum of any chord length $\boldsymbol{l_j}$ from the minima of two run length of $\boldsymbol{2^n j}$ with$\boldsymbol{n_j = [log_2(l_j - 1)]}$. If we modify our chord description to contain n, and two $\boldsymbol{x}$ values $\boldsymbol{x_{1,j} = x_{min,j}}$, and $\boldsymbol{x_2 = x_{min,j} + l_j - 2^{nj}}$, we can compute the minimum of each chord by just one additional comparison per chord.

We only need to store $\boldsymbol{[log_2(l_{max} - 1)]}$ minimum values per pixel, which can be computed in $\boldsymbol{[log_2(l_{max} - 1)] - 1}$ comparisons per pixel, as shown in figure 2. Our computation for the erosion now becomes

$$\boldsymbol{(f \ominus A)(x,y) = \min_{j \in \{0,1,\ldots N_c - 1\}} (\min(W_{y+y_j}(n_j, x + x_{1,j}), \left(W_{y+y_j}(n_j, x + x_{2,j})\right)} \qquad (4)$$

In which $\boldsymbol{x_{1,j} and\ x_{2,j}}$ are the $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ values of chord j. This means we can compute any erosion for any structuring element in time and memory complexity $\boldsymbol{O(N_c + log l_{max})}$ per pixel. Figure 3 shows the pseudo code for computing the erosion on a single line.The algorithm mention above regarding image content is independent in its time complexity, unlike the method of Van Droogenbroek and Talbot [1]. Both their method and our method extend readily to 3-D. In our case we need to augment the S.E. with a$\boldsymbol{z_{min}}$ and $\boldsymbol{z_{max}}$, each chord with a $\boldsymbol{z}$-offset, and the arrays $\boldsymbol{W_{v,}}$ need to be replaced by $\boldsymbol{W_{v,z.}}$.

```
/* Compute the minima for the first chord */
for (x = 0; x < X; x++)
g(x, y) = min(Wy+y0 (n0, x + x1,0),
Wy+y0 (n0, x + x2,0));
/* Compute for all other chords */
for (j = 1; j ≤ Nc; j ++)
for (x = 0; x < X; x++)
g(x, y) = min(g(x, y),Wy+y0 (n0, x + x1,j),
Wy+y0 (n0, x + x2,j ));
```

Fig.3. Pseudo code for performing the erosion on one image line, in which the output image line is given by $g(x, y)$.

This shows that one comparison per pixel for the first chord of the S.E. is needed, and two per pixel for each next one.

II. ARCHITECTURE DIAGRAM

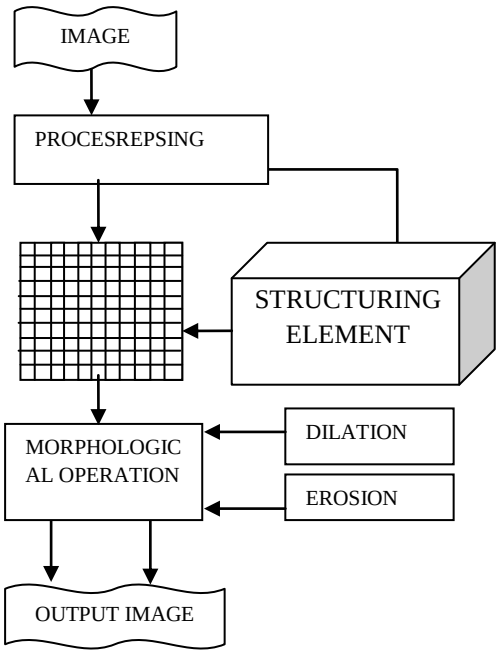
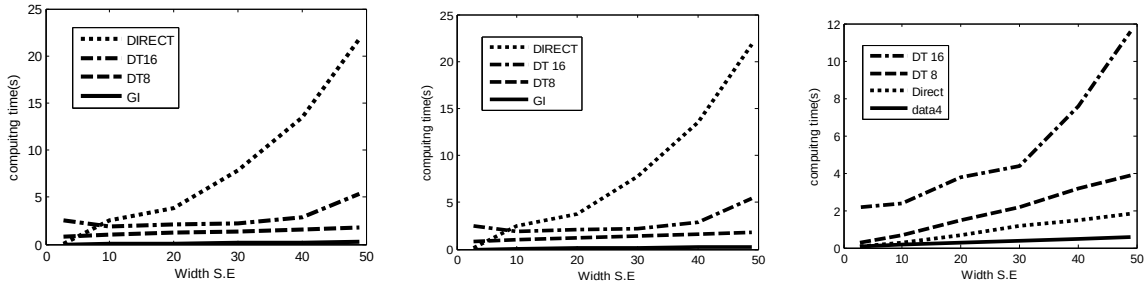


Fig. 4. Architecture diagram

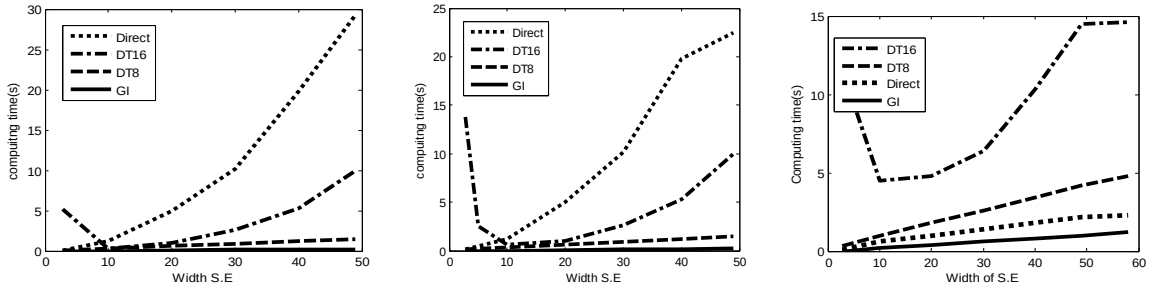
V. EXPERIMENTS

In our method we can optimise the computation time with direct implementation, here we only consider foreground pixels of the S.E. and the algorithm of Van Droogenbroeck and Talbot's [1] method using circular, H-shaped, checkerboard, rectangular, and octagon structuring elements to applied on two bit 2160 x 1440 gray scale images. For the comparison of the computation time of the native program and the program developed, we consider the foreground pixels of the image, with the usage of Van Droogenbroeck and Talbot's using the circular, check board, octagon, rectangular and H-shaped structuring elements are applied to two bit 2160 x 1440 gray scale. In this process the noise distribution in the original and the generated image are uniform. Further in order to measure the affect of gray scale numbers on the computation time of the process the original image is converted to the 16 bit version and then varying the gray scale value of the image. The Fig. 5&6 shows the computation time for the system configuration of 2.26 GHZ Intel core i3 processor based on PC, with 3 GB of RAM. A thin letter of H S.E and the circular S.E of increasing width were taken as the shapes of S.E. As discussed previously, the computation time required for the operators shape and size of the image with our method is much more reduced compared to all the other methods taken individually.



CRICLE S.E RECTANGLE S.E H-SHAPED S.E

Fig. 5 required computing time for erosions with strutting element circle(left) ,rectangular(middle), H-shaped(left) using naive method, the Van Droogenbroek- Talbot(DT8 and DT16),and our proposed (GI) algorithm on a natural image.



CRICLE S.E RECTANGLE S.E H-SHAPED S.E

Fig.6 required computing time for erosions with strutting element circle(left) ,rectangular(middle), H-shaped(left) using naive method, the Van Droogenbroek- Talbot(DT8 and DT16),and our proposed (GI) algorithm on a noise image.

To estimate the speed gain of this method called "New multi erode", the process was compared with the computation time of the existing method referred to as "DT many erode" and our method. From the fig it is very much evident that the method proposed has more execution speed compared to the existing method. For the 8-bit original image, the computation times are as follows: single erosion with a circle of diameter 49 – 2.03 sec for DT method, the computation time for the same diameter in the native method is 1.15 sec and for our method is 0.99s. A the

computation time for our methods is not dependent on the number of gray levels in the image, 16 bit and 8 bit images gives almost the same computation time. Even on the floating data only 30% speed had to be compromised. The existing method is not so dense when a very small S.Es are used with a 16-bit image, The fig refers to the same and give the reason for the long computing time for the smallest circular S.E. when the low perimeter area ratio such as circular and square are considered our method and the existing DT method gives the highest speed gain compared to the native method. The fig also shows the computation time for the thin letter shaped S.E instead of circle. The computation time for the single erosion using the letter H of width 49 on the 8-bit natural image are 3.96s, 5.51s and 2.31 for the DT method, native method and our method respectively. While the DT method needed 10.90s, for the 16-bit image. Finally our method offers a small improvement of 10 % over the existing methods called as "many-erode" version, as expected. It should be noted that the equal sized S.E with different shapes gives a reduction in the computation time instead of using the increasingly sized S.Es. the algorithm shows no change in the computing time between different images with the same size. The main difference in the computation time for the image is due to the difference in the scales being used due the difference in the computing tine for the Van Droogenbroeck and Talbot algorithm [1].

V. CONCLUSION

An advanced method was proposed with arbitrary 2-D flat structuring elements to compute morphological operations. The method which was proposed is independent of the number of grey levels in the image called computational complexity. This method has good computational performance compared to other existing methods, when S.E.s are used that cannot be easily decomposed into linear structuring elements. A single operator like the computation of the granulometries and dilation or erosion scale spaces are used on multiple S.E.s, better improvement is achieved. Afterwards the results are compared and computed once were stored in an auxiliary array, and reused for filtering of all succeeding S.E.s i.e; existing DT method arbitrary S.E.s can be used. Our advanced method performs very well compared to the existing method; mainly the applications which have the images with higher bit depth as common like medical imaging are used. Other than this improvement our advanced method can handle floating point images easily which is not possible in the DT method.

REFERENCES

[1] Marc Van Droogenbroeck and Hugues Talbot, *"Fast computation of morphological operations with arbitrary structuring elements,"* *Pattern Recogn. Lett.*, vol. 17, pp.1451–1460, 1996.

[2] S. Batman and E. R. Dougherty, *"Size distributions for multivariate morphological granulometries: Texture classification and statistical properties,"* *Opt. Eng.*, vol. 36, no. 5, pp. 1518–1529, May 1997.

[3] M. van Herk, *"A fast algorithm for local minimum and maximum filters on rectangular and octagonal kernels,"* *Pattern Recogn. Lett., vol*. 13, pp. 517–521, 1992.

[4] M. Van Droogenbroeck and M.J. Buckley, *"Morphological erosions and openings: Fast algorithms based on anchors", Journal of Mathematical Imaging and Vision*, vol. 22, pp. 121–142, 2005.

[5] L. Vincent, *"Morphological transformations of binary images with arbitrary structuring elements,"* *Image Process.*, vol. 22, pp. 3–23, Jan. 1991.

[6] H. Park and R. T. Chin, *"Decomposition of arbitrarily shaped morphological structuring elements,"* *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 17, no. 1, pp. 2–15, Jan. 1995.

[7] Erik R. Urbach, Associate Member and Michael H. F. Wilkinson, *"Efficient 2-D Gray scale Morphological Transformations with Arbitrary Flat Structuring Elements," IEEE Trans* vol. 17, no. 1, january 2008.

[8] E. R. Urbach and M. H. Wilkinson, *"Efficient 2-d grayscale dilations and erosions with arbitrary flat structuring elements,"* *Proc. Int. Conf. Image Processing*, 2006, pp. 1573–1576**.**

[9] J. Serra, *"Image Analysis and Mathematical Morphology,"* vol. 1, Academic Press, New York, 2 edition, 1982.

[10] P. T. Jackway and M. Deriche, *"Scale-space properties of the multiscale morphological dilation-erosion,"* *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, pp. 38–51, 1996.

Author Profile

Ilayaraja AP, M.Tech degree in Computer Science and Engineering from VIT university Vellore 2012, B.E degree in Computer Science and Engineering from Saraswati velu college of engineering sholingur, India in 2010

Gopinath MP, working as Assistant professor in School of Computing Science and Engineering at VIT University, Vellore, India.