A FAST ALGORITHM FOR COMPUTING MORPHOLOGICAL IMAGE PROCESSING PRIMITIVES

D.M.P. Hagyard, M. Razaz and P. Atkin *

School of Information Systems,
University of East Anglia,
Norwich, England

* Synoptics Imaging Systems Ltd,
Cambridge, England

ABSTRACT

Morphological image processing is performed by successive application of Minkowski or Hit or Miss primitive operations. A 2D structuring element is used to specify the directions in which the primitives operate. The success or failure of this type of image processing, for many real world imaging applications, is critically dependent on the efficiency with which these primitives are computed. If the direct definition of the primitives is used for their implementation, a brute force algorithm, then the time taken to complete the operation of a primitive is proportional to the number of pixels in the structuring element. This paper presents a new fast morphological transform (FMT) algorithm for computing binary morphological primitives. The computation time of the FMT is shown to be independent of the size of the structuring element used. The algorithm is compared against four other algorithms that we have implemented, namely, the brute force method and three fast frequency-domain convolution based algorithms for calculating morphological primitives. Many different comparative tests were performed, here we present some typical experimental results. In practically all the experiments, the FMT algorithm proved to be the fastest.

1. THE FMT ALGORITHM

The FMT algorithm operates on a 1D array of binary pixels. To perform the morphological operations a window, as wide as the structuring element used, is passed along the array of pixels maintaining a count of the number of on pixels underneath it. As the window starts off the image, the number of pixels underneath is initially set to zero. The window is then moved along the array by adding 1 to the pixel count if the next pixel in the array is on, and subtracting 1 from the pixel count if the pixel under the window at the rearward end is on. By modifying the count by the values of the incoming and outgoing pixels, the procedure can move the window along the array requiring only two comparison operations at each step irrespective of the length of the window. Once the window has moved the morphological operations are produced by writing back a value to the outgoing pixel. In the case of dilation, an on pixel is written back if the number of on pixels under the window is not zero, otherwise an off pixel is written. In the case of erosion, an on pixel is written back only if all the pixels under the window are on, otherwise an off pixel is written back.

The FMT algorithm can be performed on a line of pixels in approximately constant time with respect to the width of the window. The algorithm requires two operations per pixel allowing it to operate in linear time with respect to the number of pixels in the image, and can be applied in any direction which generates a line of pixels. Currently the directions used are horizontal, vertical and the two diagonals, although other directions are possible. Passing a window over each line of an image in turn is equivalent to dilating or eroding by a one-pixel wide line of the same length as the window. A line structuring element can be successively applied to an image to produce the same result as a 2D structuring element. To calculate the correct series of 1D windowing operations it is necessary to decompose the structuring element used. We have implemented procedures to scan in the four basic

directions. The FMT algorithm is therefore capable of handling all structuring element shapes which are made up from these directions such as convex symmetrical shapes comprising of horizontal, vertical and up and down sloping diagonal edges. The information required from the decomposition algorithm is the length of the edges in each of the four directions which can be found by applying an edge-following algorithm for half the boundary of the structuring element. The edge lengths in each direction are translated directly into the corresponding window lengths.

2. THRESHOLDED CONVOLUTION

We also developed a new alternative fast implementation for performing dilation and erosion based on the use of convolution [1] in order to compare and contrast with the performance of FMT algorithm. Briefly, the convolution kernel can be set up such that its value will be zero only when all the members of the kernel are over zero, or white, pixels in the image. If any of the kernel members are over non-zero, or black, pixels of the image then the values will multiply to more than zero and the result of the convolution must therefore be more than zero. By simply thresholding the convolution value so that every non-zero value is written as black, the convolution has the same result as the "hit" operation described by Serra [2]. The main idea behind this approach is therefore the following interpretation of these primitives. Dilation is expressed as

$$[f \oplus s](x) = h_\tau([f * s](x))$$

where $0 \le \tau < 1$, and $h_\tau(.)$ is a threshold operation defined by:

$$h\tau(x) = \begin{cases} 0, & x \le \tau \\ 1, & x > \tau \end{cases}$$

If the compliment of f(x) is given by $f^c(x) = 1 - f(x)$, then by the principle of duality, erosion can be defined by:

$$[f \ominus s](x) = (h_\tau([f^c * s](x)))^c$$

for every $0 \le \tau < 1$.

The thresholding is straightforward to perform and operates in O(n) time, where n is the number of pixels in the image. Computing the convolution on the other hand is the most time-consuming operation, and must be implemented efficiently. Three different fast algorithms were developed for implementing the binary morphological operations in the frequency domain. An efficient prime factor FFT [4,6] which we have developed for 3D image restoration applications [7,8] was used for calculating the Fourier transforms of the image and the structuring element. The first algorithm uses a circular convolution in the spatial domain, which is equivalent to multiplication in the frequency domain. The main condition for this algorithm to work is that the input image and the structuring element must have the same size. Once the convolution integral is evaluated, the result is thresholded to complete the specific morphological operation.

The morphological primitives such as dilation and erosion are usually performed using a structuring element that is considerably smaller than the input image, and therefore the structuring element has to be padded up with zeros to make it equal to the input image size. To overcome this problem and speed up considerably the computation time of the above circular convolution approach, two efficient algorithms, Overlap-Save and Overlap-Add methods, were implemented which take account of the small size of the structuring elements and do not require padding up with zeros. Briefly in both methods, the convolution integral is divided into small subdivisions or slices in the spatial domain. Circular convolution in frequency domain is then used to calculate the integral in each subdivision, and the separate convolution results are recombined in the spatial domain. In circular convolution, information that would be written off the end of the sample array due to the 'spreading' effect of the convolution will 'wrap-around' and appear at the beginning of the array. The management of these wrap-around errors is performed by the two methods in a slightly different manner. The Overlap-Add method pads the segment of the image so that the spreading of the image under convolution does not fall off the end of the sample and wrap-around on the output. The Overlap-Save method on the other hand uses overlapping segments of the image and discards those parts of the output image that contain wrap-around errors.

3. RESULTS AND DISCUSSION

The FMT, brute force and three fast convolution-based algorithms were all implemented on a DEC workstation running Ultrix (the speed spec mark of the system is 20), using 'c89' compiler with -O2 optimisation. All the algorithms were first tested for correctness by applying the morphological operations to a single pixel. The result of applying the series of 1 dimensional structuring elements should be the same

shape as the original 2D structuring element. This was the case for all algorithms, except the FMT when using structuring elements consisting of only diagonal edges. In the latter case, as demonstrated in Figure 1, the result was a checkerboard pattern for dilation of an image by a 5 x 5 diamond structuring element. The checkerboard pattern only appears if there is no horizontal or vertical component in the structuring element as any horizontal or vertical shift will be sufficient to fill in the holes in the image. This difficulty was easily overcome. For example, if we copy over the initial image with a dilation by a 3 x 3 diamond, and then apply the FMT algorithm with any diagonal windows shortened by 1, the result is the same as dilating the pixel by a 9x9 structuring element.

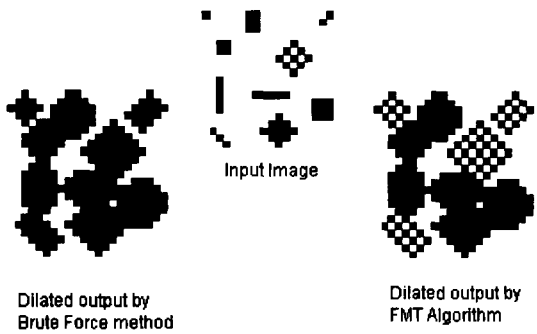


Figure 1. The effect of using the FMT to dilate an image by a 5 by 5 diamond structuring element

Next the speed of the FMT algorithm was compared with the other four methods implemented. The tests were run using a series of octagonal structuring elements ranging from 9 x 9 to 255 x 255 pixels. The input image was 864 x 864 pixels and was a greyscale image thresholded to produce an equal mixture of black and white pixels. For all the algorithms the size of the output image was altered by the morphological operations used, dilation increasing the size of the output image and erosion decreasing it. For both the Overlap-Add and Overlap-Save methods we calculated the optimum slice size such that it allowed the fastest computation of the output image. Figure 2 shows the CPU time against the structuring element size for different dilation experiments. The FMT algorithm as can be seen is the fastest, and shows a nearly flat graph, with a slight upward tilt due to the size increase of the output image. This algorithm is thus independent of the size of the structuring element, and is the fastest of all. The Overlap-Add is the next fastest method, with the Overlap-Save method being slightly slower. The convolution method without the subdivision in the spatial domain is slower still. The brute force method is the slowest of all.

Figures 3 and 4 show the corresponding results for erosion experiments. As can be seen the timing for the convolution method, without using subdivision in the spatial domain, is flat. This is because the erosion does not increase or decrease the size of the image that has to be input to the FFT algorithm. The size of image input to the FFT is the size of the input image that can be calculated in the most efficient manner. Both the Overlap-Save and Overlap-Add algorithms take more time as the structuring element size increases. This is because the slice size must remain at least twice the size of the structuring element. By the time the structuring element size is large, the slice size increases to values which make both overlap algorithms inefficient due to the extra pixels that the algorithms must process.

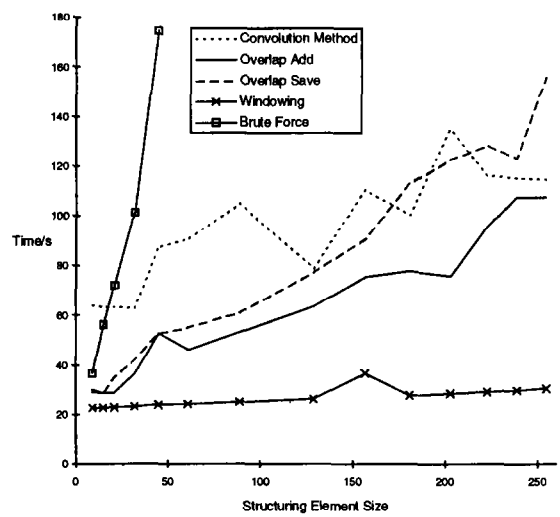
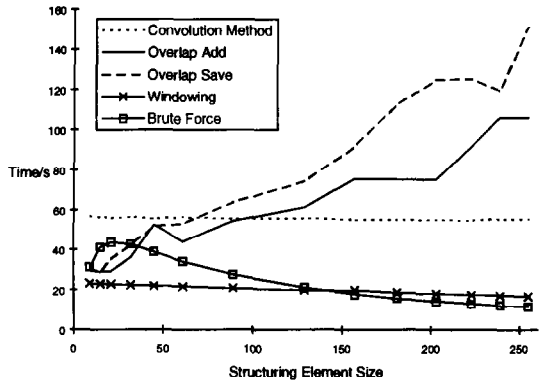


Figure 2. Graph of Time to dilate a 864 square image with an octagonal structuring element of various sizes, using 5 different dilation algorithms.

For erosion, all three convolution based methods are slower than the FMT algorithm. In this test it can be seen that the brute force algorithm is faster than the other methods for larger structuring elements. This is due to the optimisation in our algorithm that allows the brute force to stop scanning through the list of offsets for the structuring element once an off pixel has been located in the neighbourhood covered by the structuring element. For an image with an even spread of black and white pixels this means that 50% of the output pixels required only one pixel test to be shown to be blank. If a worst

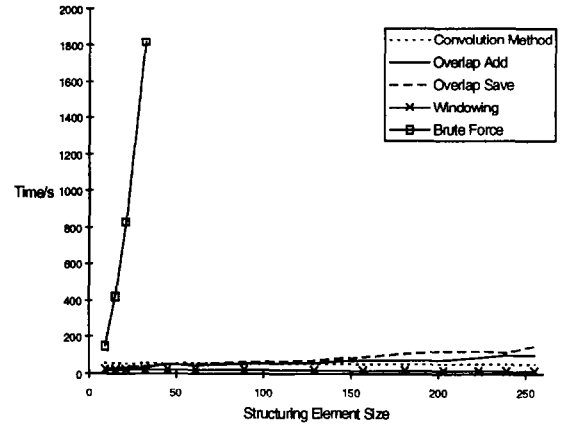
case input image is tested it can be seen that the brute force algorithm takes far longer than the other methods. Figure 4 shows the timing tests for erosion repeated with a completely black image. Here the brute force method is far slower than all the other methods.



Figue 3. Graph of Time to erode an 864 square imge, approximately mixing 'on' and 'off' pixels equally, with an octagonal structuring element of various sizes, using 5 different erosion algorithms

The results show that there is no algorithm that is superior to all others for all situations. For dilation the FMT is the fastest. This method can deal with situations where the structuring element used is convex, symmetrical and contains only vertical, horizontal and diagonal edges. For structuring elements that are more complex the Overlap Add method is the fastest. The superiority of this method compared to the convolution method drops off as the structuring element approaches the size of the input image.

For erosion with symmetric, convex structuring elements, again the FMT is the best method, however on sparse images an optimised brute force method can be surprisingly efficient. On less sparse images the brute force method is very slow indeed. With more complex structuring elements the thresholded convolution methods are fast. The Overlap Add and Overlap Save methods are more efficient when the structuring element used is significantly smaller than the input image.



Figue 4. Graph of Time to erode a 864 black square image, with an octagonal structuring element of various sizes, using 5 different erosion algorithms.

REFERENCES

[1] Kisacanin, B., Schonfeld, C., "A Fast Thresholded Linear Convolution Representation of Morphological Operations", IEEE Trans. on Image Processing, Vol. 3, No. 4, pp. 455-457, 1994.

[2] Serra, J., "Introduction to Mathematical Morphology", Computer Vision, Graphics and Image Processing, Vol. 35, pp. 283-385, 1986.

[3] Balhut, R.E., "Fast Algorithms for DSP", Addison-Wesley, 1984.

[4] DeFatta, D.J., et. al, "Digital Signal Processing: A System Design Approach", 1988.

[5] Stockham, T.G., "High Speed Convolution and Correlation", Proc. AFIPS Spring Joint Computer Conf., Vol. 28, pp. 229-233, 1966.

[6] Rader, C.M., "Discrete Fourier Transforms When the Number of Data Samples is Prime", Proc. IEEE, vol.56, pp. 1107-1108, 1968.

[7] Razaz, M., Lee, R.A., Shaw, P.J., "A Nonlinear Iterative Least Squares Algorithm for Image Restoration.", Proc. IEEE Nonlinear signal Processing, pp. 4.2-4.6, 1993.

[8] Lee, R.A., Razaz, M. et al "A Comparison of Two Nonlinear Constrained Algorithms for 3D Image Restoration", Proc. IEEE ISCAS, pp. 403-406, 1993.