EFFICIENT 2-D GRAY-SCALE DILATIONS AND EROSIONS WITH ARBITRARY FLAT STRUCTURING ELEMENTS

Erik R. Urbach and Michael H.F. Wilkinson

Institute of Mathematics and Computing Science
University of Groningen
P.O. Box 800, 9700 AV Groningen
The Netherlands

ABSTRACT

An efficient algorithm is presented for the computation of gray-scale morphological operations with 2-D structuring elements (S.E.). The required computing time is independent of the image content and of the number of gray levels used. For circular S.E.s, it always outperforms the only existing comparable method, which was proposed by Van Droogenbroeck and Talbot [1], by a factor between 1.8 and 8.6, depending on the image type. So far, filtering using multiple S.E.s is always done by performing the operator for each size and shape of the S.E. separately. With our method filtering with multiple S.E.s can be performed by a single operator for a reduced computational cost per size or shape, which makes this method more suitable for use in granulometries, dilation-erosion scale spaces, and template matching using the hit-or-miss transform.

Index Terms— Morphological operators, erosions, dilations, structuring elements, mathematical morphology, image filtering, erosion-dilation scale spaces

1. INTRODUCTION

Morphological operators [2] like dilation and erosion with structuring elements (S.E.) are the most fundamental operators in mathematical morphology and have become common tools for both image filtering and analysis [3] of binary and gray-scale images since the development of efficient algorithms [4, 5]. Usually, these algorithms are limited to linear S.E. and shapes like rectangles that can be decomposed in a series of linear S.E.s. A recent overview of efficient algorithms for morphological operators with linear S.E. and 2-D S.E. decompositions can be found in [6]. All methods based on decomposition of 2-D S.E.s into linear S.E.s share the same limitation: many shapes either cannot be decomposed efficiently or they cannot be decomposed at all. 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 [7, 8] of circles usually are used instead. For larger circles these approximations tend be either too coarse or too computationally intensive, since the number or linear S.E.s required is proportional to the diameter of the circle. Van Droogenbroeck and Talbot [1] proposed an efficient algorithm for computing morphological operations with arbitrary 2-D shapes using a histogram, which makes the computing time of their algorithm dependent on the number of gray levels used. Their idea is to compute for one pixel p of the image the complete histogram based on the intensities of the pixels around p corresponding to elements of the S.E. The value of p after erosion is the minimum intensity in the histogram which has a value > 0. For all succeeding pixels of the image (by moving around the S.E. over the image), the histogram is efficiently updated 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 (brighter) intensity which is now represented in the histogram.

In this paper, we present a new method 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 [2, 3] and erosion-dilation scale spaces [9]. Compared to Van Droogenbroeck and Talbot's method, it has the further advantage that it also works on floating point data. Our method is compared with the method of Van Droogenbroeck and Talbot in section 3.

2. ALGORITHM

For the sake of simplicity and clarity, we limit our discussion here to discrete 2-D gray-scale images f and erosions with 2-D flat S.E.s B. It should be noted that our method can be easily adapted for 3-D images and other morphological opera-

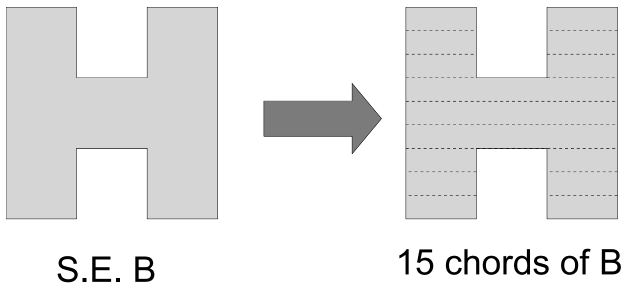

Fig. 1. Decomposition of S.E. B into chords

tions. It is assumed that all images f have their origin top-left and that images are processed in scanline order.

Like existing methods, our approach to improve the computational efficiency of erosions is by reducing the number the redundant comparisons performed by a direct implementation of its definition:

$$(f \ominus B)(p) = \min_{z \in B} f(p+z) \tag{1}$$

Our algorithm decomposes an arbitrary S.E. into a series of chords, i.e., runs of foreground pixels of maximum extent, as demonstrated in Fig. 1 for a letter H. Each chord is represented by a triple containing: (i) its y-offset with respect to the origin of the S.E., (ii) its minimal x-position, and (iii) its length l. For each S.E. we store an array of N_c chords, the minimum and maximum y-offsets $y_{\min}$ and $y_{\max}$, and minimum and maximum x-values $x_{\min}$ and $x_{\max}$, and the maximum chord length $l_{\max}$ occurring in B.

Obviously, if we can compute the minimum value within each chord, for each shifted S.E., and compute the minima of all chord-minima, we have computed the minimum of all pixels within the shifted S.E. One way to do this when processing image row y, is by creating an auxiliary 2-D array $M_{y'}(i,x)$ for each image row y' between $y - y_{\min}$ and $y - y_{\max}$. Each array index i runs from 0 to $l_{\max}$, inclusive, and x from $x_{\min}$ to $X + x_{\max} - 1$, inclusive, with X the X dimension of image f. Each value $M_{y'}(i,x)$ is defined as

$$M_{y'}(i,x) = \min_{x' \in [x, x+i]} f(x', y') \tag{2}$$

Each array can be computed in $l_{\max}$ comparisons per pixel. We can now compute the minimum over B in $N_c - 1$ further comparisons per pixel. For each value of x we compute

$$(f \ominus B)(x,y) = \min_{j \in \{0,1,\ldots,N_c-1\}} M_{y+y_j}(l_j, x + x_{\min,j}), \tag{3}$$

in which y_j, $x_{\min,j}$, and l_j denote the y-offset, minimum x position and length l of chord j. This version has a computational complexity per pixel of $O(l_{\max} + N_c)$. However, its memory complexity is quite high $O(l_{\max}(y_{\max} - y_{\min})X)$, because computing arrays M'_y only needs to be done once for every value of y'.

```
/* Copy image data into $M_{y'}(0,:)$*/
for ($x=0$; $x<X$; $x++$)
  $M_{y'}(0,x) = f(x,y')$;

/* Pad copied data on either side of
copied row */
for ($x=x_{\min}$; $x<0$; $x++$)
  $M_{y'}(0,x) = f(0,y')$;

for ($x=X$; $x<X+x_{\max}-1$; $x++$)
  $M_{y'}(0,x) = f(X-1,y')$;

/* Compute minima of runs of length $2^n$,
starting at each $x$ */
for ($i=1$; $i \leq \lfloor \log_2(l_{\max}-1) \rfloor$; $i++$)
  for ($x=x_{\min}$; $x<X+x_{\max}-1-2^{i-1}$; $x++$)
    $M_{y'}(i,x) = \min(M_{y'}(i-1,x), M_{y'}(i-1,x+2^{i-1}))$;
```

Fig. 2. Pseudocode for computing $M_{y'}$ in time proportional to $\log(l_{\max})$

We can reduce both complexities using the following observation. We can compute the minimum of any chord length l_j from the minima of two (possibly overlapping) run lengths of 2^{n_j} with $n_j = \lfloor \log_2(l_j - 1) \rfloor$. If we modify our chord description to contain n, and two x values $x_{1,j} = x_{\min,j}$, and $x_2 = x_{\min,j} + l_j - 2^{n_j}$, we can compute the minimum of each chord by just one additional comparison per chord. We only need to store $\lfloor \log_2(l_{\max} - 1) \rfloor$ minimum values per pixel, which can be computed in $\lfloor \log_2(l_{\max} - 1) \rfloor - 1$ comparisons per pixel, as shown in Figure 2. Our computation for the erosion now becomes

$$\begin{aligned}(f \ominus B)(x,y) = \\ \min_{j \in \{0,1,\ldots,N_c-1\}} (\min(M_{y+y_j}(n_j, x + x_{1,j}), \\ M_{y+y_j}(n_j, x + x_{2,j}))),\end{aligned} \tag{4}$$

in which $x_{1,j}$ and $x_{2,j}$ are the x_1 and x_2 values of chord j. This means we can compute *any* erosion (or by duality dilation) for any structuring element in time and memory complexity $O(N_c + \log l_{\max})$ per pixel.

Furthermore, any structuring element $B' \subseteq B$ can be computed using the same precomputed values, which could yield further improvements when, e.g., computing erosion-dilation scale spaces. If we have N_B nested S.E. with maximal chord length $l_{\max,B}$ and numbers of chords $N_{c,b}$ the time complexity becomes $O(\log l_{\max,B} + \sum N_{c,b})$, rather than a time complexity of $O(\sum \log l_{\max,b} + \sum N_{c,b})$, which represents a modest gain. Figure 3 shows the pseudocode for computing the erosion on a single line.

It is obvious that the algorithm described here is independent in its time complexity of the image content, 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 $z_{\min}$ and $z_{\max}$, each chord

```
/* Compute the minima for the first
chord */
for ($x = 0$; $x < X$; $x++$)
  $g(x,y) = \min(M_{y+y_0}(n_0, x + x_{1,0}),$
                 $M_{y+y_0}(n_0, x + x_{2,0}));$

/* Compute for all other chords */
for ($j = 1$; $j \leq N_c$; $j++$)
  for ($x = 0$; $x < X$; $x++$)
    $g(x,y) = \min(g(x,y), M_{y+y_0}(n_0, x + x_{1,j}),$
                   $M_{y+y_0}(n_0, x + x_{2,j}));$
```

Fig. 3. Pseudocode 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.

with a z-offset, and the arrays $M_{y'}$ need to be replaced by $M_{y',z'}$. Otherwise the extension is straightforward.

3. EXPERIMENTS

We compared the computation time of our method with a optimized naive implementation, where only the foreground pixels of the S.E. are considered, and with the algorithm of Van Droogenbroeck and Talbot [1] using circular and H-shaped structuring elements on two 8-bit 2160x1440 gray-scale images: a natural image and a generated image with uniformly distributed noise. In order to measure the influence of the number of gray levels on the computation time, 16-bit versions of these two images were computed by scaling the gray values in both images. The C source code of both algorithms is available on request.

Fig. 4 shows the computation times of these methods for a 2.8 GHz Pentium 4 processor based PC, with 1 GB of RAM. A thin letter H and circular S.E.s of increasing width were used as S.E. shapes. As noted in the previous section, when an image is to be eroded with multiple S.E.s, our approach provides a way to compute these in less time than would be required when each of them would be computed separately. To measure the speed gain of this strategy (called "New multi erode" in Fig. 4), it was compared with the total time needed for computing the same erosions using the existing [1] (referred to as "DT many erode") and our method ("New many erode").

From the figures it is clear that the method proposed is always faster than the existing methods. On the 8-bit natural image, the computation time was, for a single erosion with a circle of diameter 49, 2.03s for the DT method, 10.41s for the naive method, and 1.15s for our new method. On the 8-bit noise image the computation times for the same S.E. were respectively 2.14s, 10.14s, and 1.15s, while the DT method needed 9.89s for the 16-bit version of the same image and S.E. Since the computation time of our method is independent on the number of gray levels in the image, 16-bit and 8-bit images are computed equally fast. Even on floating point data, only a 30 % speed penalty was incurred. The histogram used by the existing method becomes extremely sparse when very small S.E.s are used with a 16-bit image, which explains the long computing time for the smallest circular S.E.s shown in Fig. 4. Both the proposed and the existing DT method have the highest speed gain compared with the naive implementation when S.E. shapes with a low perimeter area ratio are used, such as circles and squares. The computation times when a thin letter H shaped S.E. was used instead of circle is also shown in Fig. 4. The computation times for a single erosion using that letter H of width 49 on the 8-bit natural image are 3.96s, 3.31s, and 5.51s for respectively the DT method, our new method, and the naive implementation, while the DT method needed 10.90s for the 16-bit version of the same image. Finally, our "multi-erode" approach currently offers only a small improvement of about 10 % over the "many-erode" version, as expected from the differences in complexity. It should be noted that when a set of equally sized S.E.s with different shapes is used instead of the increasingly sized S.E.s used here, a stronger reduction in computing time is to be expected.

Finally, our algorithm showed no change in computing time between different images of the same size. The apparent differences between the plots of Fig. 4 are due to different scales being used due to differences in computing time for the Droogenbroeck and Talbot algorithm.

4. CONCLUSIONS

A new method for computing morphological operations with arbitrary 2-D flat structuring elements was proposed. It has a computational complexity that is independent of the number of gray levels in the image. The proposed method has a clear computational performance advantage over existing methods when S.E.s are used that cannot be easily decomposed into linear structuring elements. A further improvement is achieved when multiple S.E.s are used with a single operator, such as the computation of granulometries and dilation or erosion scale spaces [9], since the results of many comparisons are computed once and stored in an auxiliary array, which can be reused for filtering of all succeeding S.E.s.

Like the existing method of Van Droogenbroeck and Talbot [1] arbitrary S.E.s can be used. However, our method always outperforms the existing method, especially when images with higher bit depth as common in applications such as medical imaging, are used. Besides, our method can handle floating point images easily, which is impossible in the Van Droogenbroeck and Talbot method. The Van Droogenbroeck and Talbot method can however be easily adapted to other percentiles.

We are currently working on relatively small speed improvements of the method proposed by further reducing the

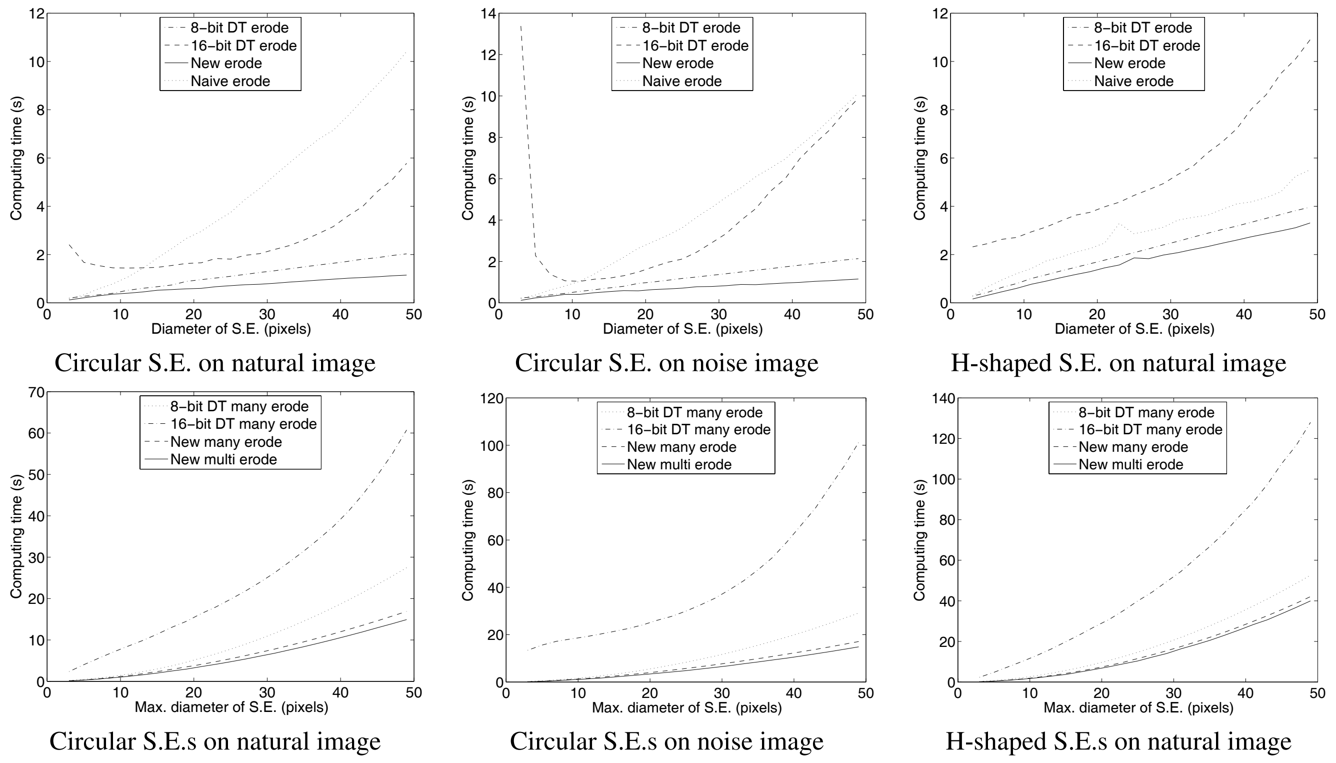

Fig. 4. Required computing time for erosions with circular (left,middle) and H-shaped (right) S.E.s using the slow naive, the Van Droogenbroeck-Talbot (DT), and our proposed (new) algorithm on a natural image and on a noise image. Top row: single erosions with width k; bottom row: series of erosions with width $l = 3, 5, 7, ..., k$.

number of comparisons needed. Further comparisons to the Droogenbroeck and Talbot, and other algorithms, on a wider range of S.E. types are also being studied, especially for the application of template matching using the hit-or-miss transform, where large sets of S.E. shapes at many scales and orientations can be computed more efficiently

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

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

[4] 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.

[5] J. Gil and R. Kimmel, "Efficient dilation, erosion, opening and closing algorithms," *IEEE Trans. on PAMI*, vol. 24, no. 12, pp. 1606–1617, 2002.

[6] 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.

[7] P. Soille, E. Breen, and R. Jones, "Recursive implementation of erosions and dilations along discrete lines at arbitrary angles," *IEEE Transactions on Pattern Analysis and Machine intelligence*, vol. 18, no. 5, pp. 562–567, May 1996.

[8] P. Soille and H. Talbot, "Directional morphological filtering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 1313–1329, 2001.

[9] 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.