FAST IMPLEMENTATIONS OF MORPHOLOGICAL OPERATIONS USING FAST FOURIER TRANSFORM (FFT)

by Olga Kosheleva[1], Sergio D. Cabrera[1], Glenn A. Gibson[1], and Misha Koshelev[2]

Departments of [1]Electrical & Computer Engineering and [2]Computer Science
University of Texas at El Paso, El Paso, TX 79968
emails {olga,cabrera}@ece.utep.edu, mkosh@cs.utep.edu

Abstract. Mathematical morphology is a methodology of processing binary (black-and-white) images. Morphological operations have a clear geometric meaning: e.g., most of them can be described via appropriate Minkowski sums. There exist many algorithms for computing Minkowski sum and therefore, for computing the morphological operations; however, in some cases, these algorithms require too much time: for $n \times n$ images, they sometimes require n^4 computational steps.

In this paper, we propose a new algorithm that requires only $O(n^2 \cdot \log(n))$ steps. This new algorithm is not geometric: it is based on using Fast Fourier Transform. Whether a purely geometric algorithm can be as fast is an open problem.

Morphological operations: what are they. Mathematical morphology was invented in the 1970s to process binary (black-and-white) images (see, e.g., [Matheron 1975], [Serra 1982], [Heijmans 1994], [Goutsias 1997], [Nguyen 1997]). Each such image can be described by the *set* of its white points.

One of the main problems of image processing is "cleaning" the image that is corrupted by noise. For black-and-white images, random noise leads to accidental white spots in black areas and accidental black spots in white areas. To delete accidental white points, we can "erode" the image, by deleting all white points that are close (e.g., ε-close, for some $\varepsilon > 0$) to black ones, and then reinserting back all the points that are ε-close to the eroded image. One can easily see that this combination of erosion and "dilation" often achieves the desired result of eliminating the noise.

From the mathematical viewpoint, the second stage (dilation) is nothing else but a Minkowski sum $E \oplus B_\varepsilon$ of the eroded set $E \subseteq R^2$ and the ball $B_\varepsilon = \{x \mid |x| \le \varepsilon\}$ of radius ε with a center in 0. Here,

$$A \oplus B = \{a + b \mid a \in A, b \in B\}. \tag{1}$$

An erosion $\ominus$ can be defined in a similar way if we take into consideration that, according to our description, the erosion $A \ominus B$ of a set A means the same thing as dilation of its complement A^c; namely:

- the complement $(A \ominus B)^c$ can be defined as the dilation $A^c \oplus B$, and therefore,
- the erosion itself can be defined as

$$A \ominus B = (A^c \oplus B)^c. \tag{2}$$

Strictly speaking, this definition only works if the set B is symmetric. In some applications, we need to use Minkowski sum for non-symmetric sets B. For general (not necessarily symmetric) sets B, the erosion is defined as follows:

$$A \ominus B = (A^c \oplus -B)^c.$$

There are many other applications of these and similar operations to processing black-and-white images.

Let us give an example of such a problem, in which we want to check whether a given set A is convex or not. It is known that convex objects on a mammogram are most probably benign, while non-convex objects have a high probability of being malignant. One way to check convexity is by using the fact that a set A is convex if and only if $0.5 \cdot (A \oplus A) = A$, where $0.5 \cdot A$ denotes $0.5 \cdot A = \{0.5 \cdot a \mid a \in A\}$ (for details, see [Popov 1997] and [Popov 1997a]).

How morphological operations are computed now. Most morphological operations can be reduced to dilation (which is Minkowski sum) and erosion, which can also be easily expressed in terms of Minkowski sum. So, in order to compute morphological operations, we must be able to compute the Minkowski sum of the two sets A and B. With image processing in mind, we can assume that both A and B are (finite) subsets of a rectangular $n \times n$ lattice (grid) L.

For such sets, the most *straightforward algorithm* for computing the Minkowski sum $A \oplus B$ is to compute the sum $a + b$ for each pair (a, b), where $a \in A$ and $b \in B$. This algorithm consists of $(\#A) \cdot (\#B)$ additions, where $\#A$ and $\#B$ denote the number of elements in the sets A and B. The largest number of elements which each set can contain is n^2, so in the worst case, this algorithm requires $n^2 \cdot n^2 = n^4$ computational steps. Even for rather crude, TV-quality images, for which $n = 512$, this number is too huge: we need $512^4 \approx 6 \cdot 10^9$ computational steps.

There exist better algorithm that work much faster for reasonable sets A and B (see, e.g., [Vincent 1993] and references therein or [Dougherty 1994]).

For example, if B is a square of linear size k, i.e., if $B = S_k$, then we can use the fact that

$$S_k = S_1 \oplus S_1 \oplus \ldots \oplus S_1 \;\; (k \text{ times})$$

and compute $A \oplus B$ as

$$A \oplus S_k = (\ldots((A \oplus S_1) \oplus S_1) \oplus \ldots \oplus S_1) \;\; (k \text{ times}).$$

Each dilation by S_1 requires $\approx n^2$ steps, and thus, the whole computation takes $k \cdot n^2 \leq n \cdot n^2 = n^3 \ll n^4$ steps.

However, no algorithm was known that would enable us to use $\ll n^4$ steps and compute $A \oplus B$ for *all* pairs A and B. Such an algorithm is proposed in this paper.

The proposed algorithm: the main idea. Each set $X \subseteq L$ can can be uniquely described by its *characteristic function*, i.e., a function $\chi_X(x)$ which is equal to 1 if $x \in X$ and 0 if $x \notin X$. So, if we have two sets A and B, this means that we have two membership functions $\chi_A(x)$ and $\chi_B(x)$. A typical image processing operation of *convolution* is defined as follows:

$$(\chi_A * \chi_B)(x) = \sum_t \chi_A(t) \cdot \chi_B(x - t).$$

Each value $(\chi_A * \chi_B)(x)$ is a sum of non-negative numbers and therefore, non-negative. This value is positive if and only if at least one term in the sum is positive, i.e., if there exists a point t for which $t \in A$ and $x - t \in B$. In other words, the value $(\chi_A * \chi_B)(x)$ is positive if and only if x can be represented as the sum of two points $t \in A$ and $x - t \in B$. Recalling the definition of the Minkowski sum, we can conclude that $(\chi_A * \chi_B)(x) > 0$ if and only if $x \in A \oplus B$.

Thus, if we can compute the convolution fast, we will be able to compute the Minkowski sum fast too. The following two facts are known:

- first, the Fourier transform of the convolution of two functions is equal to the product of their Fourier transforms;
- second, both the Fourier transform and the inverse Fourier transform of a function defined by its $N = n^2$ values can be computed in time $O(N \cdot \log(N)) = O(n^2 \cdot \log(n))$ (see, e.g., [Oppenheim 1989], [Cormen 1990], [Van Loan 1992]); the corresponding algorithms are called *Fast Fourier Transform* (FFT, for short).

In view of these two facts, we arrive at the following algorithm:

The algorithm. Given the characteristic functions $\chi_A(x)$ and $\chi_B(x)$, we do the following:

- use the FFT algorithm to compute the Fourier transform $A(\omega)$ and $B(\omega)$ of the characteristic functions $\chi_A(x)$ and $\chi_B(x)$;
- compute $C(\omega) = A(\omega) \cdot B(\omega)$;
- use the FFT algorithm to compute the inverse Fourier transform $c(x)$ of the function $C(\omega)$ (i.e., the convolution $c(x)$ of the characteristic functions $\chi_A(x)$ and $\chi_B(x)$);
- finally, for each $x \in L$, we can compute the value of the characteristic function $\chi_{A \oplus B}(x)$ as $\chi_{A \oplus B}(x) = \text{sign}(c(x))$, i.e.:
  - if $c(x) > 0$, then $\chi_{A \oplus B}(x) = 1$;
  - if $c(x) = 0$, then $\chi_{A \oplus B}(x) = 0$.

In the worst case, this algorithm is indeed faster. The above algorithm requires $O(n^2 \cdot \log(n)) + O(n^2) + O(n^2 \cdot \log(n)) + O(n^2) = O(n^2 \cdot \log(n))$ computation steps, which is indeed much faster than $O(n^4)$.

Numerical experiments using the FFT package from [Davies 1995] show that this algorithm is indeed faster for $n \geq 16$ (for $n = 32$, the new algorithm is almost 10 times faster than the straightforward algorithm, and it is even faster for $n = 512$; similar experimental results are described in [Kosheleva 1996] and [Kosheleva 1997]).

Other morphological operations can be also computed fast. We have shown that FFT helps to compute *dilations* in

$$O(n^2 \cdot \log(n))$$

steps. This leads to the possibility of fast computation of other morphological operations.

For example, we can use the formula (2) to compute the *erosion*. This formula reduces the erosion to a dilation and two complements. Since we represent a set X by its characteristic function $\chi_X(x)$, computing a complement means computing, for each $x \in L$, the value $\chi_{X^c}(x) = 1 - \chi_X(x)$, which takes $\#L = n^2$ steps. Thus, computing the erosion of the given two sets requires $O(n^2 \cdot \log(n)) + 2n^2 = O(n^2 \log_2(n))$ steps.
Other morphological operations can be represented as combinations of dilation and erosion:

- an *opening* is an erosion followed by a dilation;
- a *closing* is a dilation followed by erosion,
- etc.

Each such combination can be, therefore, also computed in time $O(n^2 \cdot \log(n))$.

Caution: for small B, the straightforward algorithm can be faster. The new algorithm for the Minkowski sum is faster only for the *worst* case, when both sets A and B are large. In a frequent case when the set B is small, the straightforward algorithm requires the time $(\#A) \cdot (\#B) \leq \mathrm{const} \cdot n^2$ which is, for small $\#B$, smaller than $n^2 \cdot \log(n)$. Thus, for small B, the straightforward algorithm is faster.

(For larger sets A and B, e.g., in mammographic applications, when we compute $A \oplus A$, the FFT-based algorithm is faster.)

Open problem. This new algorithm is not geometric: it is based on using Fast Fourier Transform. Whether a purely geometric algorithm can be as fast is an interesting open problem.

Acknowledgments. The authors are thankful to Daniel E. Cooke, Edward R. Dougherty, Vladik Kreinovich, Antony T. Popov, and Scott A. Starks for important reprints and valuable discussions.

References

Th. H. Cormen, Ch. L. Leiserson, R. L. Rivest, *Introduction to algorithms*, MIT Press, Cambridge, MA, 1990.

R. Davies, `Newmat08a`*, a matrix library in* C++. `http://nz.com/webnz/robert/index.html`.

E. R. Dougherty and D. Sinha, "Computational mathematical morphology", *Signal Processing*, 1994, Vol. 38, pp. 21–29.

J. Goutsias, R. P. S. Mahler, and H. T. Nguyen (eds.), *Random Sets: Theory and Applications*, Springer-Verlag, N.Y., 1997.

H. J. A. M. Heijmans, *Morphological image operators,* Academic Press, Boston, 1994.

O. Kosheleva, S. D. Cabrera, G. A. Gibson, and M. Koshelev, "Fast Implementations of Fuzzy Arithmetic Operations Using Fast Fourier Transform (FFT)", *Proceedings of the 1996 IEEE International Conference on Fuzzy Systems*, New Orleans, September 8–11, 1996, Vol. 3, pp. 1958–1964.

O. Kosheleva, S. D. Cabrera, G. A. Gibson, and M. Koshelev, "Fast Implementations of Fuzzy Arithmetic Operations Using Fast Fourier Transform (FFT)", *Fuzzy Sets and Systems*, 1997, Vol. 91, No. 2, pp. 269–277.

G. Matheron, *Random sets and integral geometry*, Wiley, New York, 1975.

H. T. Nguyen and V. Kreinovich, "From Numerical Intervals to Set Intervals", *Reliable Computing*, 1997, Vol. 3, No. 1, pp. 95–102.

A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, Prentice Hall, Englewood Cliffs, NJ, 1989.

A. T. Popov, “Convexity indicators based on fuzzy morphology”, *Pattern Recognition Letters*, 1997, Vol. 18, No. 3, pp. 259–267.

[Popov 1997a] A. T. Popov, “On some properties of convexity indicators based on fuzzy morphology”, *SPIE Proceedings Vol.* 3168 – *Vision Geometry VI*, 1997.

J. Serra, *Image analysis and mathematical morphology*, Academic Press, London, 1982.

C. Van Loan, *Computational Frameworks for the Fast Fourier Transform*, SIAM, Philadelphia, 1992.

L. Vincent, “Morphological algorithms”, in: E. Dougherty (ed.), *Mathematical Morphology in Image Processing*, Marcel Dekker, NY, 1993, pp. 255–288.