**RC 21197 (log #94685) 1JUN98 Engineering and Technology 6 pages**

# Research Report

## Discrete Morphological Filters for Metrology

Vijay Srinivasan
IBM Research Division
T.J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

**LIMITED DISTRIBUTION NOTICE**

This report has been submitted for publication outside of IBM and will probably be copyrighted if accepted for publication. It has been issued as a Research Report for early dissemination of its contents. In view of the transfer of copyright to the outside publisher, its distribution outside of IBM prior to publication should be limited to peer communications and specific requests. After outside publication, requests should be filled only by reprints or legally obtained copies of the article (e.g., payment of royalties).

IBM Research Division
Almaden T.J. Watson Tokyo Zurich Austin

# DISCRETE MORPHOLOGICAL FILTERS FOR METROLOGY

*Vijay Srinivasan*
IBM T.J.Watson Research Center & Columbia University
New York, U.S.A.

*Abstract*: Morphological filters have been used successfully in image processing for the past few decades. Now they are beginning to be applied in the field of computational surface metrology. Morphological filters are closely related to envelope filters. A recently formed ISO group (ISO/TC 213/AG 9) is studying various filtering techniques, including morphological filters, in order to assess their suitability for adoption as international guidelines and standards. As a part of this study discrete morphological filters for processing metrological data have been defined and their computability has been demonstrated. This paper defines morphological filters under consideration and presents explicit MATLAB implementations of discrete versions of these filters. Two examples from metrology are illustrated.

*Keywords*: metrology, filters, morphology

## 1 INTRODUCTION

In computational surface metrology raw data acquired from a probe is digitally filtered to remove noise and unnecessary details. One of the popular digital filters is the Gaussian filter. As the field of surface metrology advanced, more and general digital filters were proposed and used successfully in a variety of applications in industry. It soon became clear that some standardization in the classification, definition, and use of such digital filters in metrology was needed. This became the task of an international group of experts assembled in ISO/TC 213/AG 9 under the leadership of Paul Scott from the U.K. For the past two years this group has been studying various filtering techniques used or being considered by industry for surface metrology. One of these techniques is morphological filtering. This paper describes progress made thus far in this ISO group in demonstrating the computability of a discrete version of morphological filters.

General morphological filters are introduced in Section 2 through Minkowski sums of sets. These are specialized to discrete representations of univariate functions (signals) in Sections 3 where MATLAB implementations of algorithms for discrete morphological filtering are presented and two examples are illustrated. Section 4 concludes the paper with some discussion and remarks about future work.

## 2 MORPHOLOGICAL FILTERS

The genesis of morphological filters is Minkowski sums. If $A$ and $B$ are two sets in $\Re^n$ then their Minkowski addition is denoted and defined as $A \oplus B = \{a + b : a \in A, b \in B\}$. Minkowski addition is also known as the vector addition of sets. It can be viewed geometrically as the translational sweep of one of the sets by the other. It can be seen that the Minkowski addition is commutative, that is, $A \oplus B = B \oplus A$. Minkowski subtraction is defined and denoted as $A \ominus B = \overline{\overline{A} \oplus B}$ where overbar denotes complementation.

Next come several morphological operations which are defined in terms of Minkowski sums. Here it is customary to call one of the sets, say $A$, as the input set and the other, $B$, as the structuring element. A symmetric version of $B$ is obtained by a reflection of $B$ through the origin and is denoted $\check{B} = \{-b : b \in B\}$. With these terms and notations, we have the following two primary morphological operations[1]:

**Dilation:** Dilation of $A$ by $B$ is $D(A,B)=A \oplus \check{B}$.

**Erosion:** Erosion of A by $B$ is $E(A,B)=A \ominus \check{B}$.

Dilation "expands" the input set by the structuring element, while erosion "shrinks". Applying dilation

and erosion consecutively two secondary morphological operations are obtained as:

**Opening:** Opening of *A* by *B* is $O(A,B)=D(E(A,B), \check{B})$.

**Closing:** Closing of *A* by *B* is $C(A,B)=E(D(A,B), \check{B})$.

It is possible to apply these operations further in sequence to obtain other complex morphological operations.

In image processing literature morphological operations defined above are referred to as morphological filters. In binary morphology the input image *A* and the structuring element *B* are treated as sets in $\Re^2$. In gray-scale morphology the filtering operations are invoked on functions defined over a domain in $\Re^2$. For details the reader is referred to [2,3]. A mathematical link between morphological operations on sets and functions can be established through "umbra" transformations [3]. Informally, the umbra of a function is the set of points "under" the graph of the function. Without elaborating the details, which can be found in the references quoted above, we simply observe that morphological filtering of functions of *n* variables can be shown to be equivalent to the corresponding morphological operation on sets in $\Re^{n+1}$. As a general rule, morphological operations on sets should be taken as the fundamental mathematical definitions because they are more general, and morphological filtering of functions should be derived and justified from the set definitions.

For metrological filtering applications we will follow the convention established in signal and image processing. This means that we will speak of morphological filters for functions of one and two variables. For simplicity of exposition, morphological filters for metrology are first developed for functions of one variable (also called "signals") so that they can be applied and tested for surface profiles. These are the filters discussed in this paper. These can be extended conceptually to functions of two variables that cover the case of real surfaces.

In general, signal processing deals with both analog (continuous) and digital (discrete) signals. Lately digital signal processing has received far greater attention due to the immense popularity, availability and proven utility of digital computers. We will follow this trend and focus on discrete morphological filters.

## 3 DISCRETE MORPHOLOGICAL FILTERS

A signal is represented as a vector **x** of finite size *n*. This can be interpreted as a discrete representation of a univariate function with a sampling interval *T*. Then $x_i$, the $i^{th}$ component of **x**, is the value of the function evaluated at *iT*. Outside the finite domain of interest the function is assumed to drop down to negative infinity. Following this convention, the structuring element is denoted as a vector **b** of fixed size and is interpreted similarly. Note that **x** and **b** have the same sampling interval *T*. A discrete morphological filter takes **x** and **b** as input and produces a signal **y** of size *n* as the output and will be interpreted as stated above. Algorithm 1 describes a MATLAB function that generates a circular disk (ball) structuring element. Taking advantage of the symmetry of the disk, only the first quadrant is discretely generated. Algorithm 2 does the same for a horizontal line segment (flat) structuring element. Here only the right half of the line segment is generated.

```
function b=ballel(radius,delta)
%
% A ball shaped structuring element
% MATLAB function M-file
% radius is the radius of the ball.
% delta is the spacing for discretizing the ball.
% b is a row vector, containing the first quadrant
%     of the circle bounding the ball (i.e., the disk).
%
mmax = fix(radius/delta) + 1;
for m = 1 : mmax   % generate "first quadrant" ball points
  b(m)= sqrt( radius^2 - ((m-1)*delta)^2 );
end
```

Algorithm 1. Generating a ball structuring element.

```
function b=hlsegel(hlen,delta)
%
% A horizontal line-segment structuring element
% MATLAB function M-file
% hlen is half the length of the line-segment.
% delta is the spacing for discretizing the line-segment.
% b is a row vector, containing the right half of
%    the full line-segment.
%
mmax = fix(hlen/delta)+ 1;
for m = 1 : mmax   % generate "right half" line-segment points
  b(m)= 0.0;
end
```

Algorithm 2. Generating a flat structuring element.

There are several discrete morphological filters. The discrete counterparts of dilation, erosion, opening and closing have been defined in the image and signal processing literature, for example in [2]. Algorithms 3 and 4 present MATLAB function M-files to dilate and erode functions. In these algorithms the structuring element is assumed to be symmetric about both the horizontal and vertical axes. This condition implies that the structuring element is symmetric about the origin.

The basic idea behind Algorithms 3 and 4 is to position the origin of the structuring element at every point of the input and summing them. Extreme values at each sampling point are collected and reported as the output. Notice how the end conditions are handled. The only basic computational units involved in these algorithms are *shift* by array indexing, *add/subtract* of integers and reals, and *compare* using max/min functions. A proof of correctness of these algorithms may appear to be intuitively obvious. But it involves several subtle assumptions and is not presented here due to space limitation. It is instructive to observe the similarity between these algorithms and those based on ray-representation of solids [5].

```
function y = f_dilate(x,b)
%
% Function dilation - MATLAB function M-file
% y is a function dilation of x by b.
% x and b are row vectors, and so is y.
%
nmax = size(x,2); % size of row vector x
mmax = size(b,2); % size of row vector b
%
% First, process the interior points.
%
for n = mmax : nmax-mmax+1
  y(n) = x(n) + b(1); % initialize
  for m = 2 : mmax  % iterate to obtain “convolution”
    y(n) = max( y(n) , x(n+m-1)+b(m) );
    y(n) = max( y(n) , x(n-m+1)+b(m) );
  end
end
%
% Next, process sets of end points on both sides.
%
for m = 1 : mmax-1
  y(m) = x(m) + b(1); % initialize left end points
  y(nmax-m+1) = x(nmax-m+1) + b(1); % initialize right end points
  for k = 1 : m  % iterate to obtain end-half “convolutions”
    y(m) = max( y(m) , x(m-k+1)+b(k) );
    y(nmax-m+1) = max( y(nmax-m+1) , x(nmax-m+k)+b(k) );
  end
  for k = 2 : mmax % iterate to obtain the other half “convolutions”
    y(m) = max( y(m) , x(m+k-1)+b(k) );
    y(nmax-m+1) = max( y(nmax-m+1) , x(nmax-m+2-k)+b(k) );
  end
end
```

Algorithm 3. Dilating a function by a structuring element.

```
function y = f_erode(x,b)
%
% Function erosion - MATLAB function M-file
% y is a function erosion of x by b.
% x and b are row vectors, and so is y.
%
nmax = size(x,2); % size of row vector x
mmax = size(b,2); % size of row vector b
%
% First, process the interior points.
%
for n = mmax : nmax-mmax+1
  y(n) = x(n) - b(1); % initialize
  for m = 2 : mmax  % iterate to obtain "convolution"
    y(n) = min( y(n) , x(n+m-1)-b(m) );
    y(n) = min( y(n) , x(n-m+1)-b(m) );
  end
end
%
% Next, process sets of end points on both sides.
%
for m = 1 : mmax-1
  y(m) = x(m) - b(1); % initialize left end points
  y(nmax-m+1) = x(nmax-m+1) - b(1); % initialize right end points
  for k = 1 : m  % iterate to obtain end-half "convolutions"
    y(m) = min( y(m) , x(m-k+1)-b(k) );
    y(nmax-m+1) = min( y(nmax-m+1) , x(nmax-m+k)-b(k) );
  end
  for k = 2 : mmax % iterate to obtain the other half "convolutions"
    y(m) = min( y(m) , x(m+k-1)-b(k) );
    y(nmax-m+1) = min( y(nmax-m+1) , x(nmax-m+2-k)-b(k) );
  end
end
```

Algorithm 4. Eroding a function with a structuring element.

Algorithms 5 and 6 describe MATLAB functions to close and open functions by a structuring element by simply applying the dilation and erosion operations in succession. Figure 1 illustrates how an input signal is filtered morphologically by a ball structuring element. Figures 2 shows the results when the structuring element is changed to a flat. In both figures, the input signal is uniformly sampled at 0.5 μm interval. In general, dilation and closing produce signals that are *above* the input signal (extensivity); erosion and opening result in signals that are *below* the input (antiextensivity) [2,3].

```
function y=f_close(x,b)
%
% Function closing - MATLAB function M-file
% Function closing of x by b.
% x and b are row vectors, and so is y.
%
y = f_erode(f_dilate(x,b),b);
```

Algorithm 5. Closing a function with a structuring element.

```
function y=f_open(x,b)
%
% Function opening - MATLAB function M-file
% Function opening of x by b.
% x and b are row vectors, and so is y.
%
y = f_dilate(f_erode(x,b),b);
```

Algorithm 6. Opening a function with a structuring element.

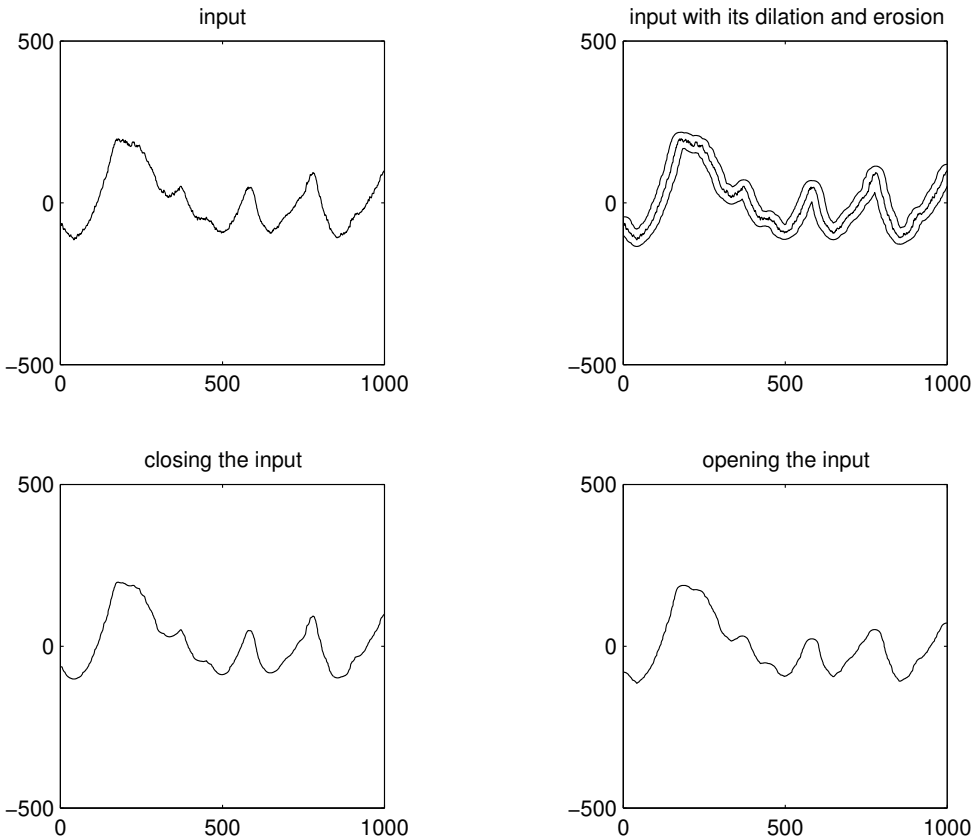


Figure 1. Discrete dilation, erosion, closing and opening of a signal by a circular disk of 20 μm radius. All dimensions are in μm.

## 4 DISCUSSION AND CONCLUDING REMARKS

The time complexities of the algorithms for discrete dilation, erosion, closing, and opening presented in the last section are all linear in the size of the input signal (size of the vector x). They also take only linear space for computation. Therefore these algorithms are *asymptotically optimal*. The only restriction on the structuring element is that it be symmetrical about a vertical and a horizontal axis. This is a reasonable assumption for all practical structural elements have this property. The algorithms can be easily modified if this assumption is violated without affecting their time or space complexity. These algorithms are also very robust because the only computations involved are shift (by array indexing), add and compare.

We have thus demonstrated that digital morphological filters are computable and that the associated algorithms are as competitive as the conventional Gaussian filtering algorithms. Morphological filters can clearly take on the role of envelope filters; they may even be applied for a wider class of problems. If **x** represents the locus of the center of a circular probe representable by **b**, then f_erode(**x**,**b**) is a representation of the underlying real surface profile.

Algorithms in this paper are presented as MATLAB function M-files. Therefore they are "ready to run". Figures reported in this paper were generated by just running these MATLAB functions. The reader can test them on his/her data to see if their performance and output are satisfactory. They can be reimplemented easily in lower level languages like C. There is a general consensus among ISO technical experts in favor of developing and distributing a public MATLAB toolbox for morphological and other filters for metrology. This paper takes a small initial step in that direction.

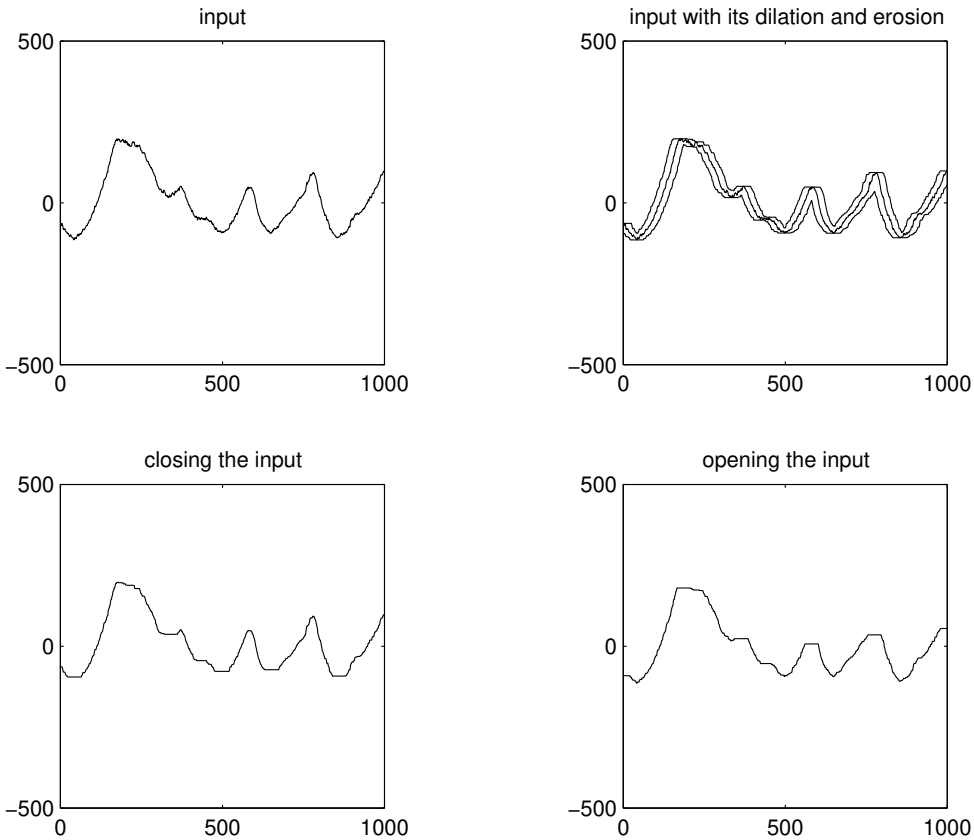


Figure 2. Discrete dilation, erosion, closing and opening of a signal by a horizontal line segment of 40 μm length. All dimensions are in μm.

There are at least two directions in which the work reported in this paper can be extended. The first is to process bivariate functions to handle surfaces. The second is to handle profiles that are not represented as functions, for example, a circular profile.

## ACKNOWLEDGMENT AND A DISCLAIMER

The author would like to acknowledge many of his ASME and ISO colleagues who offered valuable advice and encouragement for the work reported in this paper. However, opinions expressed in this paper are his own and they do not reflect the official position ISO or any of its member bodies.

## REFERENCES

[1] G. Matheron, *Random sets and integral geometry*, John Wiley & Sons, New York, 1975.
[2] J. Serra (Editor), *Image analysis and mathematical morphology*, Vols. 1 and 2, Academic Press, San Diego, 1988.
[3] E. R. Dougherty, *An introduction to morphological image processing*, SPIE Optical Engineering Press, Bellingham, 1992.
[4] G. Strang and T. Nguyen, *Wavelets and filter banks*, Wellesley-Cambridge Press, Wellesley, 1997.
[5] J.P.Menon, R.J.Marisa and J.Zagajac, *More powerful solid modeling through ray representations,* IEEE CG&A, Vol. 14, No. 3, pp. 22-35, May 1994.

Contact point: Room 2-150, IBM T.J.Watson Research Center, P.O.Box 218, Yorktown Heights, NY 10598, U.S.A. Phone 914-945-3411, Fax 914-945-3242, E-mail: vasan@watson.ibm.com