FAST COMPUTATION OF MORPHOLOGICAL AREA PATTERN SPECTRA

Arnold Meijster

Centre for High Performance Computing
University of Groningen, P.O. Box 800
9700 AV, Groningen, The Netherlands

Michael H. F. Wilkinson

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

ABSTRACT

An area based counterpart of the binary structural opening spectra is developed. It is shown that these area opening and closing spectra can be computed using an adaptation of Tarjan's union-find algorithm. These spectra provide rotation, translation, and scale invariant pattern vectors for texture analysis.

1. INTRODUCTION

Multiscale feature extraction from textures or images can be done by means of size distributions or granulometries [1, 2]. These are ordered sets of morphological openings which remove details below given scales. Usually the scale is defined by the width of features, though other measures are possible [3, 4]. One method to summarize the action of a size distribution on a particular image in a single, 1-D array is called a pattern spectrum [5, 6]. In the binary case, this summarizes how many foreground pixels remain as a function of the scale of the openings used. Figure 1 shows an example of a binary image and two morphological pattern spectra. Image details are present at those scales where the slope of the spectrum is steep.

Fast algorithms for the binary case are available [7, 8], but computing patterns spectra in the grey scale case can be time consuming, because many types of granulometries require repeated openings and summing of grey levels to compute the spectrum. Though computationally efficient algorithms exist for certain openings with particular classes of structuring elements [9], granulometries using Euclidean discs to achieve rotation invariance of the method are still costly. Breen and Jones [4] have shown that spectra based on attribute openings, which are connected set filters can be computed efficiently using a pixel-heap based algorithm. Attribute openings are extensions of *area* operators which form a comparatively new class of image processing operators [3]. They filter out or highlight detail based on the area, rather than the width of an object. In the case of attribute openings, the filters can be based on a far larger class of attributes of each connected component. Rotation invariant attribute openings are easily designed, so rotation invariant spectra can be obtained quickly. In this paper we propose a new algorithm, based on Tarjan's union-find method. This method is a variant of earlier work on area and attribute openings [10, 11]. We will focus on the case of area spectra, though the method is easily extended to other attributes.

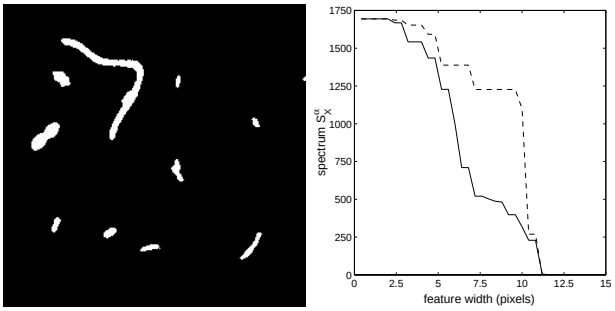

Fig. 1. A binary image and two morphological pattern spectra derived from granulometries based on structural opening by discs of increasing sizes (solid), and openings-by-reconstruction with similar structuring elements (dashed).

In this paper we first describe opening and closing spectra, and the definitions of area openings and closings. We then go on to define area pattern spectra, and describe the new algorithm for their computation. Finally we give some experimental results of computational efficiency.

2. THEORY

2.1. Opening and Closing Pattern Spectra

Let binary images X and Y be defined as the set of foreground pixels, i.e., a subset of the image domain $\mathbf{M} \subset \mathbb{R}^n$ (usually $n = 2$). Grey scale images are a mapping from $\mathbf{M}$ to $\mathbb{R}$. A size distribution or granulometry is a set of operators $\{\alpha_r\}$ with r from some ordered set Λ (usually $\Lambda \subset \mathbb{R}$

or $\mathbb{Z}$), with the following three properties

$$\alpha_r(X) \subseteq X \tag{1}$$
$$X \subseteq Y \quad \Rightarrow \quad \alpha_r(X) \subseteq \alpha_r(Y) \tag{2}$$
$$\alpha_r(\alpha_s(X)) = \alpha_{\max(r,s)}(X), \tag{3}$$

in the binary case, and in the grey scale case

$$\alpha_r(f) \leq f \tag{4}$$
$$f \leq g \quad \Rightarrow \quad \alpha_r(f) \leq \alpha_r(g) \tag{5}$$
$$\alpha_r(\alpha_s(f)) = \alpha_{\max(r,s)}(f), \tag{6}$$

for all $r, s \in \Lambda$. Since (3) and (6) imply idempotence, it can be seen that size distributions are openings. The pattern spectrum S_X^α obtained by applying a size distribution α_r to binary image X is defined as

$$S_X^\alpha(r) = A(\alpha_r(X)). \tag{7}$$

where $A(X)$ is a function denoting the Lesbesgue measure of set X in $\mathbb{R}^n$, or the number of pixels in $\mathbb{Z}^n$. In the grey-scale case the spectrum $S_f^\alpha(r)$ of image f is just the integral of the grey level of $\alpha_r(f)$ over the image domain, or in the discrete case the sum:

$$S_f^\alpha(r) = \sum_{x \in \mathbf{M}} (\alpha_r(f))(x). \tag{8}$$

2.2. Area Openings and Closings

The theory of area operators is given only briefly here. For a more thorough discussion the reader is referred to [3]. Here we will first discuss binary area openings and closings, and then the extension to the grey scale case. Binary area openings are based on binary connected openings. Let the set $X \subseteq \mathbf{M}$ denote a binary image with domain $\mathbf{M}$. The binary connected opening $\Gamma_x(X)$ of X at point $x \in \mathbf{M}$ yields the connected component of X containing x if $x \in X$, and $\emptyset$ otherwise. Thus Γ_x extracts the connected component to which x belongs, discarding all others.

The binary area opening can now be defined as:

Definition 1 *Let $X \subseteq \mathbf{M}$ and $r \geq 0$. The binary area opening of X with scale parameter r is given by*

$$\Gamma_r^a(X) = \{x \in X | A(\Gamma_x(X)) \geq r\} \tag{9}$$

The binary area closing can be defined by duality:

$$\Phi_r^a(X) = [\Gamma_r^a(X^c)]^c \tag{10}$$

The definition of an area opening of a grey scale image f is most easily derived from binary images $T_h(f)$ obtained by thresholding f at h, which can be defined as:

$$T_h(f) = \{x \in \mathbf{M} | f(x) \geq h\} \tag{11}$$

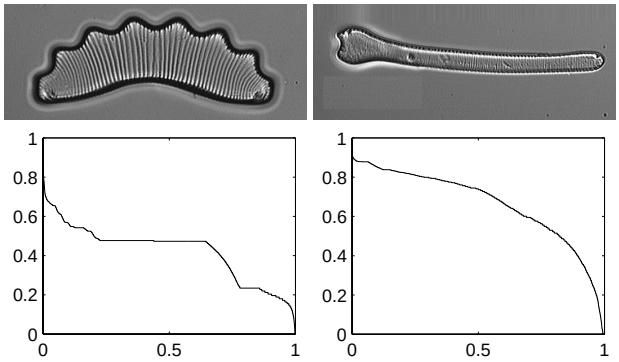

Fig. 2. An example of area pattern spectra: (top): images of diatoms showing fine internal details; (bottom): corresponding pattern spectra obtained from series of area openings with increasing size criterion. Axes have been normalized

Definition 2 *The area opening for a mapping $f : \mathbf{M} \longrightarrow \mathbb{R}$ is given by:*

$$(\gamma_r^a(f))(x) = \sup\{h | x \in \Gamma_r^a(T_h(f))\}. \tag{12}$$

The grey scale area closing ϕ_r^a is defined as:

$$(\phi_r^a(f))(x) = \inf\{h | x \in \Phi_r^a((T_h(f))^c)\}, \tag{13}$$

The interpretation of these definitions is that the area opening of an image assigns each point the highest threshold at which it still belongs to a connected foreground component of area r or larger. The area closing assigns each point the lowest threshold at which it belongs to a connected background component of area r or larger.

Vincent [3] has shown that area openings and closings are size distributions and anti-size distributions respectively. Therefore, area opening and closing spectra can be defined as in (8). Fig. 2 shows an example of such spectra.

3. COMPUTING AREA PATTERN SPECTRA

Let N_s be the number of scales we wish to use for our pattern spectra. A naive implementation would compute N_s openings and compute the sum of grey levels for each opening. This is very costly for even modest values of N_s. We therefore propose a new algorithm, which processes the image just once, rather than $O(N_s)$ times.

Before going into the details of the algorithm, we first define a *flat zone* L_h at level h of a grey scale image f as a connected component of the set of pixels $\{p \in \mathbf{M} | f(p) = h\}$. A *regional maximum* M_h at level h is a flat zone no members of which have neighbors larger than h. A *peak component* P_h at level h is a connected component of $T_h(f)$. At each level h there may be several such sets, which will be indexed as L_h^i, P_h^j and M_h^k, respectively, with i, j, and k

from some index set. It can be seen that any regional maximum M_h^k is also a peak component, but the reverse is not true. Finally, let Λ be the set of values the attribute (in this case area) of the peak components can assume for a given image size, and B be a mapping from Λ to $[0..N_s)$, which is the index range of the spectrum, stored in array `spec`.

The idea is to process the image in grey scale order, filling peak components at all grey levels starting at the regional maxima. Each time we move to a new grey level h we may find some $P_{h'}^j$ at grey level $h' > h$ which must be merged with some L_h^k. In this case, `spec`$[B(A(P_{h'}^k))]$ must be incremented by $(h'-h)A(P_{h'}^k)$, because an area opening of $r = A(P_{h'}^k)$ would remove this quantity from the sum of grey levels at that point in the image.

To make this work we need an efficient way to keep track of the peak components, which are of course disjoint sets. Tarjan [12] presents the union-find algorithm which provides a general method for keeping track of disjoint sets. It allows performing set-union operations on sets which are in some way equivalent, while ensuring that the end product of such a union is disjoint from any other set. Tarjan uses tree structures to represent sets. Each non-root node in a tree points to its parent, while the root is flagged in some way. Two objects x and y are members of the same set if and only if x and y are nodes of the same tree, which is equivalent to saying that they share the same root. There are three important operations.

- `Makeset(x)`: Create a new singleton set $\{x\}$.
- `FindRoot(x)`: Return the root element of the set containing x.
- `Union(x,y)`: Compute the union of the two sets containing x and y.

Each of the sets is represented as a tree, with each pixel containing a pointer to its parent pixel. To store the trees for the entire image, we use an integer array `parent` of the same size as the image `I` (i.e. N), in which `parent[p]` is the parent of pixel `p`. Pixels are stored as `width*y+x`,with `x` and `y` the pixel's x and y coordinates, and `width` the image width. If a pixel is a root of a tree, i.e. it has no parent, we flag this by setting `parent[p] = -area`, with `area` the number of pixels in the set. In the following discussion pixel `p` refers to the integer representation of the coordinates, whereas `I[p]` is the pixel's grey level.

Finally, we need to ensure that for a peak component P_h at level h, its root element `r` has a grey level `I[r]`$= h$. Therefore, we always make the last pixel processed the root of the new tree. This is done by radix-sorting the pixels, and storing the coordinates in an array `S` of length N. Pixels of the same grey level are processed in scan line order.

As can be seen in Figure 3, The `Union` procedure performs the following task: Find the root `r` of neighbor `n`;

```
void MakeSet ( int x )
{ parent[x] = -1;
}

void Link ( int x, int y )
{ parent[y] = parent[y] + parent[x];
  parent[x] = y;
}

int FindRoot ( int x )
{ if ( parent[x] >=0 )
    { parent[x] = FindRoot( parent[x] );
      return parent[x];
    }
  else return x;
}

void Union ( int n, int p )
{ int r=FindRoot(n);
  if ( r != p )
    { if ( I[p]<>I[r]
        { spec[B(-parent[r])] -=
            (I[r]-I[p]) * parent[r];
        }
      Link( r, p );
    }
}
```

Fig. 3. The basic operations for area spectra. The root element now stores the area of the component as a negative number.

`FindRoot(p)` is not necessary because the current pixel `p` is always a root, because if linking has occurred it has become the root of the resulting tree, and otherwise it is a singleton. If the root `r`$\neq$`p` we must `Link` them. Before linking we check whether `I[r]`$\neq$ `I[p]`, and if so, update the appropriate bin in `spec`.

The complete algorithm is shown in Fig. 4. The `spec` array is first set zero, after which the pixels are processed in reverse grey level order. For each pixel `p` a union with those neighbours which have already been processed is performed. After this stage `spec[i]` is the amount subtracted from the sum of grey levels at the i^{th} scale, but not at lower scales. The desired spectral value S_i is given by

$$S_i = S_{i-1} - \texttt{spec[i]} \quad \text{if } i > 0 \tag{14}$$

For $i = 0$ the correct value is the sum of grey levels minus `spec[0]`. This is carried out by the last loop in Fig. 4.

```
/* array S contains sorted pixel list */
for (i=0;i<Ns;i++)
  spec[i]=0;
greysum=0;
for (p=0; p<Length(S); p++)
  {
    pix = S[p];
    greysum += I[p];
    MakeSet(pix);
    for all neighbors nb of pix do
      if ((I[pix] < I[nb]) ||
          ((I[pix] == I[nb]) && (nb<pix)))
        Union(nb,pix);
  }
for (i=0;i<Ns;i++)
  { spec[i] = greysum - spec[i];
    greysum = spec[i];
  }
```

Fig. 4. Pseudo-code showing how to compute an area opening pattern spectrum using the operations of Fig. 3.

Table 1. Timing results of different algorithms (in seconds)

Method	min	median	max
naive,$N_s = 256$	23.0	30.7	33.3
pixel-heap	0.14	0.68	1.42
union-find	0.092	0.124	0.127

4. RESULTS

Three algorithms were implemented in C: (i) the naive implementation using fast area openings by union-find with $N_s = 256$ (ii) the direct approach using the pixel-heap method (for details see [4]), and (iii) the direct implementation using union-find. All implementation were optimized manually for maximum performance. Timings were performed on 10 natural images of 256×256 pixels. The results are shown in Table 1. As expected, the direct approaches outperform the naive by a factor approaching N_s. The union-find approach outperforms the pixel-heap method by a factor of about 5, consistent with earlier findings on attribute openings [11]. Similar results were obtained on images of 768×1024 (data not shown).

5. DISCUSSION

We have shown that the new algorithm is up to ten times faster than the pixel-heap method on natural images. The complexity of the algorithm is identical to that of a single area or attribute opening using union-find, which was shown to be $O(N \log N)$, whereas that of the pixel-heap method is $O(N^2 \log N)$, with N the number of pixels [10, 11]. This method can readily be extended to other attribute operators, in the same way as in [11] for openings and closings.

6. REFERENCES

[1] G. Matheron, *Random Sets and Integral Geometry*, John Wiley, 1975.

[2] L. Vincent, "Granulometries and opening trees," *Fundamenta Informaticae*, vol. 41, pp. 57–90, 2000.

[3] L. Vincent, "Morphological area openings and closings for grey-scale images," in *Shape in Picture: Mathematical Description of Shape in Grey-level Images*, Y.-L. O, A. Toet, D. Foster, H. J. A. M. Heijmans, and P. Meer, Eds., pp. 197–208. NATO, 1993.

[4] E. J. Breen and R. Jones, "Attribute openings, thinnings and granulometries," *Computer Vision and Image Understanding*, vol. 64, no. 3, pp. 377–389, 1996.

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

[6] P. Maragos, "Pattern spectrum and multiscale shape representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 701–715, 1989.

[7] P. F. M. Nacken, "Chamfer metrics in mathematical morphology," *Journal Mathematical Imaging and Vision*, vol. 4, pp. 233–253, 1994.

[8] P. F. M. Nacken, "Chamfer metrics, the medial axis and mathematical morphology," *Journal Mathematical Imaging and Vision*, vol. 6, pp. 235–248, 1996.

[9] L. Vincent, "Fast grayscale granulometry algorithms," in *ISMM'94, Mathematical Morphology and its Application to Image Processing*, J. Serra and P. Soille, Eds., Fontainebeau, France, 1994, pp. 265–272, Kluwer Academic Publishers.

[10] A. Meijster and M. H. F. Wilkinson, "An efficient algorithm for morphological area operators," Tech. Rep. 99-9-07, Institute for Mathematics and Computing Science, University of Groningen, submitted.

[11] M. H. F. Wilkinson and J. B. T. M. Roerdink, "Fast morphological attribute operations using Tarjan's union-find algorithm," in *Proceedings of the ISMM2000*, Palo Alto, CA, June 2000, pp. 311–320.

[12] R. E. Tarjan, "Efficiency of a good but not linear set union algorithm," *Journal of the ACM*, vol. 22, pp. 215–225, 1975.