# Texture Segmentation by Multiscale Aggregation of Filter Responses and Shape Elements

Meirav Galun Eitan Sharon* Ronen Basri† Achi Brandt‡
The Weizmann Inst. of Science
Dept. of Computer Science and Applied Math.
Rehovot 76100, Israel

## Abstract

*Texture segmentation is a difficult problem, as is apparent from camouflage pictures. A Textured region can contain texture elements of various sizes, each of which can itself be textured. We approach this problem using a bottom-up aggregation framework that combines structural characteristics of texture elements with filter responses. Our process adaptively identifies the shape of texture elements and characterize them by their size, aspect ratio, orientation, brightness, etc., and then uses various statistics of these properties to distinguish between different textures. At the same time our process uses the statistics of filter responses to characterize textures. In our process the shape measures and the filter responses crosstalk extensively. In addition, a top-down cleaning process is applied to avoid mixing the statistics of neighboring segments. We tested our algorithm on real images and demonstrate that it can accurately segment regions that contain challenging textures.*

## 1 Introduction

Camouflage is a striking evidence that segmentation can be hard not only for computers. Animal fur, painted carefully to match the color and texture of its habitat, provides a hiding place for a predator and a safe refuge for a prey. Computer systems often have difficulties to properly segment even simpler images. In particular these systems face difficulties when the image includes complex textures. This paper presents a novel approach to texture segmentation. Our approach manages to accurately segment real images that contain challenging textures.

We approach the problem of texture segmentation using a framework that combines structural characteristics of texture elements with filter responses. Our process adaptively identifies the shape of texture elements and characterize them by their size, aspect ratio, orientation, brightness, etc., and then uses various statistics of these properties to distinguish between different textures. For these statistics, the texture elements need not be identical, or made of uniform shade of gray. Instead, they may vary in shape and size, and their spacing may be irregular. Texture elements are identified at multiple scales, and their statistics (densities of various element properties) are used to influence the segmentation process, including the identification of larger texture elements, allowing for example to identify texture elements that themselves are textured. At the same time our process uses the statistics of filter responses to characterize textures. Most importantly, these responses can be averaged while avoiding mixing the statistics of neighboring textures. The two components, the shape measures and filter responses crosstalk extensively during the algorithm. The shape of texture elements determine what filter responses are considered relevant. At the same token filter responses may affect the formation of texture elements.

The importance of properties such as size, aspect ratio, orientation, brightness, and density of repeated texture elements was noted already in the classical perceptual studies of Julesz [7] and Beck [1]. However, only few attempts (e.g., [15]) were made to directly use these principles in computational studies due to their limited applicability to complex, natural textures. A popular alternative approach has been to characterize texture by measuring their response to filter banks [10, 16, 9]. Filter responses characterize the frequency, size and orientation of the texture elements, but without explicitly isolating them. (An interesting recent variant attempts to identify the "textons" by clustering the filter outputs [8].)

*Current address: Dept. of Applied Math, Brown University

†Research was supported in part by the European Commission Project IST-2000-26001 VIBES and by the Israeli Ministry of Science, Grant No. 2104. The vision group at the Weizmann Inst. is supported in part by the Moross Foundation.

‡Research was supported by grants No. 295/01 from the Israel Science Foundation, US Air Force 5408-05-sc-0007, GIF 3982 and by the Carl F. Gauss Minerva Center for Scientific Computation at the Weizmann Institute of Science.The authors thank D. Ron for highlighting remarks and useful discussions.

Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003) 2-Volume Set
0-7695-1950-4/03 $17.00 © 2003 IEEE

While filter banks clearly provide a useful characterization of textures, their application poses some difficulties in texture segmentation. In particular, filter outputs along the boundaries of a segment can significantly differ from their output within the segment both because they mix the statistics of neighboring segments [13] and because boundary edges (if exist) may invoke strong response in the direction of the edge [9]. This, at best, may lead to inaccurately detecting the boundaries of segments, and in more severe cases to either missing or hallucinating segments. Adaptive scale selection [9, 4] only partially solves this problem, because some textures cannot be characterized by a single scale, such as when a texture is composed of several elements that are differently spaced, or when the elements themselves are textured.

Our multilevel method of bottom-up weighted aggregation of picture elements, which involves also some top-down iterative feedback, is different from other approaches to texture in the following three principles: (1) Mini-segments at various scales are isolated, and various statistics of their individual properties are collected and used to characterize large-scale textures. (2) Mixing of statistics between neighboring texture regions can be avoided. (3) Texture measures characterize picture aggregates at every scale (except for the few finest levels), playing a central role in defining the way these aggregates are combined to create the next-coarser-scale aggregates. These principles are incorporated into an intensity-based segmentation process [12, 13] allowing the detection of both textured and non-textured regions.

## 2 Multiscale graph partitioning

We implement our treatment of texture using the framework presented in [12, 13]. In this framework, given an image, we construct a graph in which every pixel is a node and neighboring pixels are connected by an edge. A weight is associated with each edge reflecting the contrast in the corresponding location in the image. A multiscale procedure is used to find optimal partitions of the graph. Below we describe the multiscale framework and its use for image segmentation. Our description explicitly highlights the relations of this framework to the normalized cuts algorithm, and provides a clearer intuition of the method.

Let $G = (V, W)$ be a weighted graph with nodes $v_i \in V$ and undirected, weighted edges $w_{ij} > 0$. We conveniently treat $W$ as a symmetric matrix with its diagonal elements set to zero.

To evaluate segments we define a saliency measure as follows. Every segment $S = \{v_1, v_2, \ldots, v_m\} \subseteq V$ is associated with a state vector $u = (u_1, u_2, \ldots, u_N)$, where $N = \|V\|$, and

$$u_i = \begin{cases} 1 & \text{if } v_i \in S \\ 0 & \text{if } v_i \notin S. \end{cases} \quad (1)$$

The saliency associated with $S$ is defined by

$$\Gamma(S) = \frac{\sum_{i>j} w_{ij}(u_i - u_j)^2}{\sum_{i>j} w_{ij} u_i u_j}, \quad (2)$$

which sums the weights along the boundaries of $S$ normalized by the internal weights. Segments that yield small values of $\Gamma(S)$ are considered salient[1]. In matrix notation $\Gamma$ can be written as

$$\Gamma(S) = \frac{u^T L u}{\frac{1}{2} u^T W u}, \quad (3)$$

where $L$ is the Laplacian matrix whose elements are

$$l_{ij} = \begin{cases} \sum_{k\,(k \neq i)} w_{ik} & i = j \\ -w_{ij} & i \neq j. \end{cases} \quad (4)$$

If we allow arbitrary real assignments to $u$ the minimum for $\Gamma$ is obtained by the minimal generalized eigenvector $u$ of

$$Lu = \lambda W u \quad (\text{minimal} \quad \lambda > 0). \quad (5)$$

This equation is in fact equivalent to the normalized cuts solution [14].

Our objective is to find the best partitions (0-1 assignments of $u$) for this graph. We do this recursively by executing the following procedure. The finest graph ($s$=0) is denoted by $G^{[0]} \stackrel{def}{=} G$. Given a graph $G^{[s-1]} = (V^{[s-1]}, W^{[s-1]})$, and denote by $u^{[s-1]} = (u_1^{[s-1]}, u_2^{[s-1]}, \ldots, u_{N^{[s-1]}}^{[s-1]})$ the state vector at level $s-1$, we first examine all single-node partitions. For those partitions $u^{[s-1]}$ contains 1 in one entry and 0 in all other entries, and $\Gamma$ is the ratio between the diagonal elements of $L^{[s-1]}$ and $W^{[s-1]}$ (whose values are computed recursively as is explained below). After accounting for the single-node partitions we proceed to finding partitions that in the current scale involve multiple nodes. For these partitions we seek to produce a smaller graph (that contains about half of the original nodes) whose partitions can approximate these multiple-node partitions. Denote by $u^{[s]} = (u_1^{[s]}, u_2^{[s]}, \ldots, u_{N^{[s]}}^{[s]})$ the state vector at level $s$, we seek a sparse, $N^{[s-1]} \times N^{[s]}$ matrix $P$ such that

$$u^{[s-1]} \approx P u^{[s]}. \quad (6)$$

$P = P^{[s-1]}$ is called the inter-scale interpolation matrix. Assuming that such $P$ can be found then the saliency measure $\Gamma$ can be written as

$$\frac{u^{[s-1]T} L u^{[s-1]}}{\frac{1}{2} u^{[s-1]T} W u^{[s-1]}} \approx \frac{u^{[s]T} P^T L P u^{[s]}}{\frac{1}{2} u^{[s]T} P^T W P u^{[s]}}, \quad (7)$$

[1] To avoid division by zero we will ignore single pixel partitions.

where $L = L^{[s-1]}$ and $W = W^{[s-1]}$. The right hand side of this equation determines a graph with $N^{[s]}$ nodes whose weight matrix, defined by $W^{[s]} = P^T W^{[s-1]} P$, includes the edge weights in the off-diagonal elements and internal node weights in the diagonal elements. The (generalized) Laplacian of level $s$ can be computed by $L^{[s]} = P^T L P$, as resulting from (7), or be more cheaply approximated by a relation to $W^{[s]}$ as in (4).

To build the coarse graph we should still specify how to select $P$. One popular method to produce a coarse graph is through node contraction (e.g., [6, 17]). In this method strongly connected pairs of nodes at level $s-1$ are replaced each by a single node at level $s$. This, however, may lead to premature assignments of nodes and lead to poor approximations of $u^{[s-1]}$. A better way is to follow the graph coarsening method that yields a fast multilevel solver for the eigenvalue problem (5). Such a fast solver, called Algebraic Multigrid (AMG) [3, 2], yields the following procedure. We begin by selecting a set of coarse representative nodes $V^{[s]} \subseteq V^{[s-1]} = \{1, 2, \ldots, N^{[s-1]}\}$, so that every node in $V^{[s-1]} \setminus V^{[s]}$ is strongly connected to $V^{[s]}$. A node is considered strongly connected to $V^{[s]}$ if the sum of its weights to nodes in $V^{[s]}$ is a significant proportion of its total weights. Then, we construct the inter-scale interpolation matrix, $P = P^{[s-1]}$, where for any $i \notin V^{[s]}$ $\{p_{ik}\}_{k=1}^{N^{[s]}}$ are chosen to be proportional to $\{w_{ik}^{[s-1]}\}_{k=1}^{N^{[s]}}$ such that $\sum_k p_{ik} = 1$, while for $i \in V^{[s]}$, $p_{ii} = 1$ and $p_{ij} = 0$ for all $j \neq i$.

Assuming that the original graph is only locally connected, and since every node is strongly connected to $C$, there exists such a *sparse* interpolation matrix $P$. Therefore, the weights $w_{kl}^{[s]}$ can be computed efficiently by weighted aggregation [12].

This coarsening procedure is repeated recursively, so in fact we only evaluate $\Gamma$ for single nodes at all levels. As a result of this process we obtain a full pyramid representation of the image. Nodes associated with low values of $\Gamma$ represent salient segments. The rest of the nodes can each be thought of as representing a weighted aggregate of pixels that may at a higher scale be part of a salient segment. The weight $p_{ij} = p_{ij}^{[s-1]}$ can be thought of as the probability of a node $i \in V^{[s-1]}$ to belong to an aggregate $j \in V^{[s]}$. These probabilities are compounded from level to level, so that eventually, at a high level $s$, almost every pixel (a node in $V^{[0]}$) belong with probability close to 1 to one and only one aggregate in $V^{[s]}$, while few pixels near the boundaries between aggregates may still remain undecided.

This multiscale partitioning procedure can be used for image segmentation in the following way. Given an image, we begin by constructing a 4-connected graph $G = (V, W)$, where every pixel is represented by a node $v_i \in V$, and every pair of neighboring pixels are connected with an edge with weight $w_{ij}$. The weight reflects the contrast between the two pixels $i$ and $j$, e.g.,

$$w_{ij} = e^{-\alpha |I_i - I_j|}, \tag{8}$$

where $I_i$ and $I_j$ denote the intensities of the two neighboring pixels, and $\alpha$ is a positive constant.

If we now apply the multiscale procedure to find the best partitions of the graph, we will obtain a full, irregular pyramid of graphs whose salient nodes represent the salient partitions of the graph. However, these partitions will reflect segments that are distinctive only by fine contrast between pixels, and so they may not be attractive perceptually. A popular approach to overcome this problem is to determine the weight in (8) according to the responses of filters that take into account the contrast between neighborhoods of pixels. Unfortunately, this can lead to mixing of statistics of different segments and hence over-smoothing of weights and consequently to inaccurate partitions. To solve this problem [13] proposed a method to combine multiscale measures *during* the construction of the pyramid. At each coarsening step, we first determine the next coarser graph using the weighted aggregation procedure. This will produce a graph that is roughly equivalent to the finer level graph. Then, we modify the weights in the new graph to incorporate coarser measures of differences between neighboring aggregates. This idea was demonstrated using intensity statistics (which allows to only a limited extent the handling of certain textures, see a comparison in Section 4) and boundary integrity measures computed for each of the aggregates.

## 3 Texture segmentation

Texture elements are identified at multiple scales, and their statistics are used to influence the identification of larger texture elements. The bottom-up aggregation process adaptively identifies the shape of texture elements and characterize them by their size, orientation, brightness and density. In addition, filter response statistics are used to characterize texture and affect the formation of texture elements. Conversely, the shape of texture elements determines through a top-down process which of the filter responses are relevant.

We begin with the same graph as in Section 2 and perform an aggregation process. During the aggregation process we accumulate statistics of the aggregates formed. At the first few levels (typically 3-4) we use only the intensity-related properties to determine the aggregates. From a certain level and on we use also the shape measures and the filter responses to determine the aggregates. We compare these multiscale regional properties of neighboring aggregates and modify the weights between them according to

- Given an image define a four-connected graph $G^{[0]} = (V^{[0]}, W^{[0]})$ where $V^{[0]}$ is the set of image pixels and $W^{[0]}$ is defined according to (8). Calculate, for each pixel, short line-integral responses in four directions (21).
- For s=1,2,... construct $G^{[s]}$ from $G^{[s-1]}$, as follows:
  1. Select a representative set of nodes $V^{[s]}$, such that $V^{[s-1]} \setminus V^{[s]}$ is strongly connected to $V^{[s]}$.
  2. Define $P = P^{[s-1]}$ the inter-scale interpolation matrix.
  3. Calculate $W^{[s]} \approx P^T W^{[s-1]} P$ by weighted aggregation.
  4. For each node in $V^{[s]}$
     (a) calculate first and second order moments (12) and solve (13) to find length, width and orientation.
     (b) calculate average intensity and multiscale variance (11).
     (c) accumulate average properties of sub-nodes (10).
     (d) perform a top-down process to accumulate filter responses (see section 3.3.2).
  5. modify $W^{[s]}$ according to these regional properties (24).

**Table 1.** Outline of algorithm

the result of this comparison. This allows neighboring aggregates of similar textures to merge at the next levels of the aggregation process and aggregates of different textures to stand out. Our multiscale regional properties include the average dimensions (length and width) and orientation of sub-aggregates. Filter statistics include the average response of fine, edge-shaped filters obtained at the sub-aggregates. An outline of the algorithm is given in Table 1.

### 3.1 Accumulation of regional properties

All of the measures that we apply are in fact averages of some properties within a region, and so they can be accumulated during the aggregation process. Formally, given an aggregate $k$ at scale $s$, let $\bar{Q}_k^{[r][s]}$ denote a weighted average of a certain $r$-scale property $a^{[r]} = (a_1, \ldots, a_{N^{[r]}})$, i.e.,

$$\bar{Q}_k^{[r][s]} = \frac{\sum_j c_{jk} a_j^{[r]}}{\sum_j c_{jk}}, \tag{9}$$

where $c_{jk}$ are the appropriate weights. More precisely $c_{jk} = c_{jk}^{[r][s]}$ is the element $j$ in the the $k^{\text{th}}$ column of the relevance matrix, $P^{[r]} \cdots P^{[s-1]}$, i.e., the matrix product of the interpolation matrices from fine level $r$ up to coarse level $s$. Recall that $P^{[r]}$ relates level $r$ with level $r+1$, and so on.

Practically, the average properties are calculated very efficiently, without using the explicit relation above (9), but by the following recursive rule: for an aggregate $k$ at level $s$, the average of an $r$-scale property $a^{[r]}$ can be computed directly from its sub-aggregates at level $s-1$.

We define the following row vectors

$$Q^{[r][r]} \stackrel{def}{=} a^{[r]}, \quad C^{[r][r]} \stackrel{def}{=} 1^T = (1, 1, \ldots, 1),$$

$$Q^{[r][s]} \stackrel{def}{=} a^{[r]} \cdot P^{[r]} \cdots P^{[s-1]} = Q^{[r][s-1]} P^{[s-1]},$$
$$C^{[r][s]} \stackrel{def}{=} 1^T \cdot P^{[r]} \cdots P^{[s-1]} = C^{[r][s-1]} P^{[s-1]}.$$

Note that $C_k^{[r][s]}$ in fact accumulates the number of sub-aggregates of level $r$ that compose the aggregate $k$ at scale $s$. In particular, $C_k^{[0][s]}$ is the volume of the aggregate $k$ at scale $s$ in pixel units.

Given the interpolation matrix $P^{[s-1]}$ and also $Q^{[r][s-1]}$ and $C^{[r][s-1]}$, the relevant $Q$ and $C$ for scale $s$ can be computed immediately using the recursive relations above. Then, equation (9) simplifies to

$$\bar{Q}_k^{[r][s]} = \frac{Q_k^{[r][s]}}{C_k^{[r][s]}}. \tag{10}$$

These recursion rules can be used to compute not just averages, but also variances and histograms. For example, if we set $a^{[0]}$ to be the intensities of pixels, then we obtain the average intensity of an aggregate, denoted by $\bar{I}_k^{[0][s]}$ for aggregate $k$ at level $s$. If we set $a^{[0]}$ to be the squared intensity of pixels we obtain the mean squared intensity of an aggregate that can be used to determine its variance. We can also compute the average variance of sub-aggregates by starting the accumulation with $a^{[r]}$ set to be the variances of aggregates of level $r$. This way we can characterize an aggregate $k$ at scale $s$ by a multiscale variance vector

$$\overrightarrow{\nu}_k^{[s]} = (\bar{\nu}_k^{[1][s]}, \bar{\nu}_k^{[2][s]}, \ldots, \bar{\nu}_k^{[s][s]}), \tag{11}$$

as in [13]. Similarly, if we set the property $a_j^{[r]}$ to be a vector (and hence $a^{[r]}$ to be a matrix) its accumulation would produce a histogram of values.

Computing and maintaining these multiscale measurement vectors is done in linear time in the number of image pixels. Denote the number of pixels in the image by $N$. Because at every level of the pyramid the number of nodes reduces by about half, the total number of nodes in the pyramid is about $2N$. At every level $s$ we store $O(s)$ or $O(s^2)$ measurements, but since the number of nodes decreases exponentially with $s$, whereas the number of measurements grows only polynomially, the total number of measurements is still $O(N)$.

### 3.2 Shape elements

We compute for every aggregate its length, width, and orientation. Other shape moments could be added. Then, we use these quantities to produce shape statistics for the aggregates. In particular, for every aggregate we compute the average length and width of its sub-aggregates at all finer scales. In addition, we construct an orientation histogram of the sub-aggregates at all finer scales. Below we

describe these statistics and how we accumulate them as we construct the pyramid.

The dimensions and orientation of an aggregate can be computed using the second-order principal moments of the aggregate [5]. As explained above an efficient way to compute those principal moments is from the moments of the sub-aggregates (10). In this case the property $a^{[0]}$ can be the $x$ or $y$ coordinates of the image pixels, their product, or the squared values of these coordinates. In this manner, we obtain the second-order moments, associated with the shape of an aggregate $k$ at scale $s$

$$\bar{x}_k^{[0][s]}, \bar{y}_k^{[0][s]}, \overline{x_k \cdot y}_k^{[0][s]}, \overline{x_k^2}^{[0][s]}, \overline{y_k^2}^{[0][s]}. \qquad (12)$$

Higher order moments could similarly be accumulated. A short notation for (12) will be used: $\bar{x}, \bar{y}, \overline{x \cdot y}, \overline{x^2}, \overline{y^2}$.

The two principal directions are determined by the eigenvalue system

$$Se = \omega e, \qquad (13)$$

where $S$ is the covariance matrix

$$S = \begin{pmatrix} \overline{x^2} - \bar{x}^2 & \overline{x \cdot y} - \bar{x} \cdot \bar{y} \\ \overline{x \cdot y} - \bar{x} \cdot \bar{y} & \overline{y^2} - \bar{y}^2 \end{pmatrix}. \qquad (14)$$

The corresponding eigenvalues, of the eigenvectors $e = (cos\theta, sin\theta)^T$ and $e^{\perp}$, are the squared dimensions (length and width) of the shape.

For any emerging aggregate $k$ at level $s$ the second order moments are first calculated using (10). Then (13) is solved to find the principal direction, the length and the width. We denote the length of the aggregate by $L_k^{[s][s]}$, and the width by $W_k^{[s][s]}$. The orientation of the aggregate is specified by the angle $\theta$ between the principal direction and the positive $X$-axis. To remove ambiguity we set $\theta$ so that $0 \leq \theta < \pi$. In general, $\theta$ can be reliably estimated only for elongated (anisotropic) texture elements, and so we will use $\theta$ only for aggregates whose length to width ratio exceeds a certain threshold (typically 3).

Usually, coherent texture regions are characterized by the shape of finer texture elements and their density of appearance. An efficient way to code these properties is by introducing multiscale measurement vectors that accumulate the average width and average length of all finer texture elements that compose a coarser aggregate.

For an aggregate $k$ of scale $s$ we denote by $\bar{L}_k^{[r][s]}$ the average length of its sub-aggregates of scale $r$. These averages are calculated recursively using (10), with $a^{[r]}$ being the vector of length $s$ of all the aggregates at scale $r$. Performing this computation for all $r \leq s$ we obtain a vector of average lengths,

$$\overrightarrow{L}_k^{[s]} = (\bar{L}_k^{[1][s]}, \bar{L}_k^{[2][s]}, \ldots, \bar{L}_k^{[s][s]}). \qquad (15)$$

To reflect the measure of similarity between two texture regions, represented by two aggregates $k$ and $l$ at level $s$, a normalized distance is defined

$$D(L)_{kl}^{[s]} = \left( \sum_{r=1}^{s-2} \left( \frac{\bar{L}_k^{[r][s]} - \bar{L}_l^{[r][s]}}{\bar{L}_k^{[r][s]} + \bar{L}_l^{[r][s]}} \right)^2 \right)^{1/2}. \qquad (16)$$

The normalization is due to the natural increase of the dimensions with scale. Similar description is suitable for the average width $\overrightarrow{W}_k^{[r][s]}$, yielding the corresponding normalized distance $D(W)_{kl}^{[s]}$

The orientation of elongated texture element provides an important cue in texture segmentation. We find the principal orientation of elongated aggregates (whose aspect ratio exceeds a certain threshold) using (13). We then prepare for every aggregate a histogram depicting the distribution of orientations of its sub-aggregates at all finer scales. Each aggregate $k$ at scale $s$ holds a two-dimensional $s \times n$ histogram, where $n$ is the number of direction bins. Each row $r$ in the histogram contains the number of elongated texture elements of scale $r$ in each direction bin. The histogram is filled recursively along the aggregation process, using

$$H_k^{[r,j][s]} = \sum_i p_{ik} H_i^{[r,j][s-1]}, \qquad (17)$$

which is exactly the calculation of the numerator in (10).

This accumulation process begins as follows. Given an elongated aggregate $i$ of scale $r$ whose direction is $\theta$, $0 \leq \theta < \pi$, and let $b$ be the integer, $b = 0, \ldots, n-1$, for which $\frac{\pi}{n} \cdot b \leq \theta < \frac{\pi}{n} \cdot (b+1)$. Then the entry $H_i^{[r,j][r]}$ in the histogram is determined by the linear anterpolation (the adjoint of the linear interpolation):

$$H_i^{[r,j][r]} = \begin{cases} \frac{(b+1) \cdot \pi/n - \theta}{\pi/n} & j = b \\ \frac{\theta - b \cdot \pi/n}{\pi/n} & j = b+1 \quad \text{if } b < n-1 \\ \frac{\theta + \pi/n - \pi}{\pi/n} & j = 0 \quad \text{if } b = n-1 \\ 0 & \text{otherwise.} \end{cases} \qquad (18)$$

For each aggregate $k$ at scale $s$ we count in each histogram bin $H_k^{[r,j][s]}$ how many elongated sub-aggregates of scale $r$ had their principal direction associated with bin $j$. We define a measure of distance between two aggregates $k$ and $l$ of scale $s$ as

$$D(H)_{kl}^{[s]} = \left( \sum_{j=0}^{n-1} \sum_{r=1}^{s} \left( \frac{H_k^{[r,j][s]}}{C_k^{[0][s]}} - \frac{H_l^{[r,j][s]}}{C_l^{[0][s]}} \right)^2 \right)^{1/2}. \qquad (19)$$

In this equation each bin is normalized by the volume of the corresponding aggregate so that the measure will reflect the density of the finer texture elements in each direction.

### 3.3 Filter responses

We employ a technique that can incorporate filter responses from all scales and orientations. This method integrates well with the multiscale framework and maintains the adaptive support of measurements. Since filter responses have very different values near the boundaries between different textures, an accurate way to collect filter response statistics on texture regions is to use the *interior* support of the aggregates calculated by our bottom-up process. By a top-down process, we can collect statistics only from internal filter responses, while neglecting outer filter responses. As mentioned above, this can be useful for any kind of filter at any scale. However, our experience indicates that in this context our main need is for filter responses at the finest (pixel) level, to capture fine texture elements, such as hair or grass, that may not be captured as aggregates due to lack of sufficient contrast. The details of the procedure follows.

#### 3.3.1 Fine edge filter responses

For each pixel $(i,j)$, we calculate the line integrals in four directions:

$$\begin{aligned} L^1_{i,j} &= \frac{1}{4}I_{i,j-1} + \frac{1}{2}I_{i,j} + \frac{1}{4}I_{i,j+1} \\ L^2_{i,j} &= \frac{1}{4}I_{i-1,j} + \frac{1}{2}I_{i,j} + \frac{1}{4}I_{i+1,j} \\ L^3_{i,j} &= \frac{1}{4}I_{i-1,j+1} + \frac{1}{2}I_{i,j} + \frac{1}{4}I_{i+1,j-1} \\ L^4_{i,j} &= \frac{1}{4}I_{i-1,j-1} + \frac{1}{2}I_{i,j} + \frac{1}{4}I_{i+1,j+1}, \end{aligned} \quad (20)$$

where $I_{i,j}$ denotes the intensity at pixel $(i,j)$. We use these integrals to obtain the absolute responses of edge filters:

$$\begin{aligned} F^1_{i,j} &= |L^1_{i-1,j} - L^1_{i+1,j}| \\ F^2_{i,j} &= |L^2_{i,j-1} - L^2_{i,j+1}| \\ F^3_{i,j} &= |L^3_{i-1,j-1} - L^3_{i+1,j+1}| \\ F^4_{i,j} &= |L^4_{i+1,j-1} - L^4_{i-1,j+1}|. \end{aligned} \quad (21)$$

We can compute the average of the absolute responses at any scale $s$ recursively using (10), with $a^{[0]}$ standing for one of the properties $F^d,\ d = 1,\ldots,4$ of all pixels. However, this calculation will be biased by strong filter responses at the boundaries of segments. To solve this problem we perform a top-down process that eliminates such undesired effect. This top-down process is explained below. As a result of this top-down process, we obtain for an aggregate $k$ at any scale $s$ a filter response distribution vector denoted by

$$\overrightarrow{\bar{F}}^{[s]}_k = (\bar{F}^1_k, \bar{F}^2_k, \bar{F}^3_k, \bar{F}^4_k)^{[s]}. \quad (22)$$

The similarity between two textured regions is defined by the correlation coefficient

$$D(F)^{[s]}_{kl} = \frac{\mathrm{Cov}(\overrightarrow{\bar{F}}^{[s]}_k, \overrightarrow{\bar{F}}^{[s]}_l)}{\sigma(\overrightarrow{\bar{F}}^{[s]}_k) \cdot \sigma(\overrightarrow{\bar{F}}^{[s]}_l)}. \quad (23)$$

As with shape-related measurements above, various inter-scale histograms can be accumulated based on $\bar{F}^d_k$ and similar averages obtained with larger (and perhaps wider) filter masks. Our experience (see below) has shown that even the simple similarity measures (23) already yields much improved segmentation.

#### 3.3.2 Top-down process

Filter responses have very different values near the boundaries between different textures. To eliminate this effect we perform a top-down "cleaning" process in which we eliminate responses from pixels whose relevance values are small. Specifically, for each aggregate $k$ at scale $s$ we first find the pixels that belong to $k$, or in other words the support of measurements calculated for aggregate $k$ by the bottom-up aggregation process. To do so we begin with the characteristic state vector $u^{[s]}$ which is set to 1 at the $k$'th entry and 0 elsewhere. By repeating interpolations from scale $s$ all the way down to scale 0, using (6), we obtain for each pixel its relevance value. A pixel whose relevance value is below 0.5 is considered outside the aggregate and its filter response statistics will not be taken into account. Only the pixels whose relevance values are higher than 0.5 will contribute to the statistics, and their responses will be averaged with their relevance values as weights. Extension of this top-down process to wider and longer filter masks will require to generalize the definition of outer filter responses, e.g., if "most" of the pixels in the filter mask demonstrate low relevance, this filter response should be neglected in the statistics accumulation.

This top-down process raises the linear complexity of the algorithm by a log factor. In principle we can maintain linear complexity if we bound the top-down process to go down a bounded number of levels.

### 3.4 Weights update induced by regional properties

Each edge weight $w^{[s]}_{kl}$ of two aggregates $k$ and $l$ at level $s$ is calculated by using weighted aggregation. Then, to reflect the measure of similarity between two texture regions, this weight is modified by multiplying it by

$$\begin{aligned} & e^{-\tilde{\alpha}D(I)^{[s]}_{kl}} \cdot e^{-\beta D(\nu)^{[s]}_{kl}} \cdot e^{-\rho(1-|D(F)^{[s]}_{kl}|)} \\ & \cdot e^{-\gamma(D(L)^{[s]}_{kl}+D(W)^{[s]}_{kl})} \cdot e^{-\omega D(H)^{[s]}_{kl}}, \end{aligned} \quad (24)$$

where $D(I)^{[s]}_{kl}$ and $D(\nu)^{[s]}_{kl}$ are defined as in [13]. The positive constants are explained in the next section.

## 4 Experiments

We have implemented our method and tested it on natural images. We selected a set of challenging images that contain animals in camouflage. While humans may use high-level information to segment such images, we approach this problem using data-driven, low-level processing only. For comparison we show segmentation results obtained with the algorithms described in [13] and in [9]. [13] performs weighted aggregation that includes a limited handling of texture using only brightness measures (variance of sub-aggregates, Eq. (11)). [9] combines a filter-based texture segmentation and intensity based segmentation in a normalized-cuts framework. A gating mechanism is used to overcome boundary problems. Our process in contrast does not determine a-priori whether segmentation should be based locally on texture or intensity contrast, but combines these measures uniformly throughout the image.

In our implementation, we set the parameters around $\alpha = 10$, $\tilde{\alpha} = 4$, $\beta = 0.5$, $\gamma = 3$, $\omega = 3$, $\rho = 1$ and $n = 4$. In any of the experiments below we did not apply the boundary integrity process suggested in [13]. Our application takes 5 seconds to complete the bottom-up aggregation of a $400 \times 400$ image using a Xeon $1.6$ Ghz processor. Together with the top-down cleaning process implemented to the pixel level the runtime increases to 10 seconds.

Figure 1 shows a typical set of results. The table contains (from left to right) the original images, the results obtained with our method, results obtained using [13] and [9]. In all pictures we show the original image along with an overlay of the segmentation results. In all five animal examples the animal was segmented in one piece by our method outperforming the other two algorithms. Notice in particular that our method managed to accurately segment both the leopard (third row) despite gradual variations in texture and the polar bear (fifth row) despite differences in intensities. In contrast, both [13] and [9] lead to over-fragmentation of the images and in some cases to leakage problems. The bottom image shows a natural composition of textures. Notice the accurate separation of the two parts of the brick wall in our method.

## 5 Conclusions

We have presented a novel method for texture segmentation and demonstrated that it can achieve state-of-the-art results when applied to challenging textures. We chose to characterize textures by a collection of statistics that include shape, intensity variability, and filter responses. While we generally find this set to be adequate for a large variety of textures, it is possible to incorporate additional statistics in our framework. For example, we can use higher order shape moments or variance of filter responses. One important issue is how to combine the various statistics into a single weight. We are currently exploring ways to cast this problem in a Bayesian formulation so we can learn from real images how to optimally combine the different statistics.

It is important to note that the segmentations produced by our approach are *hierarchical*, and with their associated statistics they can be used directly for recognition and retrieval, because every segment in our method comes with an identifying list of numbers describing its shape, texture, and its sub-segments along with their own descriptors.

## References


[1] J. Beck. Textural segmentation. In J. Beck, editor, *Organization and Representation in Perception*. Erlbaum, Hillsdale, N.J., 1982.

[2] A. Brandt. Algebraic multigrid theory: the symmetric case. *Appl. Math. Comput*, 19:23–56, 1986.

[3] A. Brandt, S. McCormick, and J. Ruge. Algebraic multigrid (amg) for automatic multigrid solution with application to geodetic computations. Inst. for Computational Studies, POB 1852, Fort Colins, Colorado, 1982.

[4] G. Caenen, V. Ferrari, Zalesny, A., and L. Van Gool. Analyzing the layout of composite textures. *Texture*, I:15–19, 2002.

[5] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley and Sons, New York, 2001.

[6] Y. Gdalyahu, D. Weinshall, and M. Werman. Self organization in vision: stochastic clustering for image segmentation, perceptual grouping, and image database organization. *PAMI*, 23(10):1053–1074, 2001.

[7] B. Julesz. Textons, the elements of texture perception and their interactions. *Nature*, 290:91–97, 1981.

[8] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *IJCV*, 43(1):29–44, 2001.

[9] J. Malik, S. Belongie, T. K. Leung, and J. Shi. Contour and texture analysis for image segmentation. *IJCV*, 43(1):7–27, 2001.

[10] J. Malik and P. Perona. Preattentive texture discrimination with early vision mechanisms. *JOSA*, 7A(5):923–932, 1990.

[11] P. Perona. Deformable kernels for early vision. *PAMI*, 17(5):488–499, 1995.

[12] E. Sharon, A. Brandt, and R. Basri. Fast multiscale image segmentation. *CVPR*, I:70–77, 2000.

[13] E. Sharon, A. Brandt, and R. Basri. Segmentation and boundary detection using multiscale intensity measurements. *CVPR*, I:469–476, 2001.

[14] J. Shi and J. Malik. Normalized cuts and image segmentation. *PAMI*, 22(8):888–905, 2000.

[15] H. Voorhees and T. Poggio. Computing texture boundaries from images. *Nature*, 333:364–367, 1988.

[16] T. Weldon, W. Higgins, and D. Dunn. Efficient gabor filter design for texture segmentation. *Pattern Recognition*, 29:2005–2015, 1996.

[17] D. Willersinn, E. Bertin, and W. Kropatsch. Dual irregular voronoi pyramids and segmentation. In *Technical Report*. Technical University of Vienna, March 1994.




0-7695-1950-4/03 $17.00 © 2003 **IEEE**

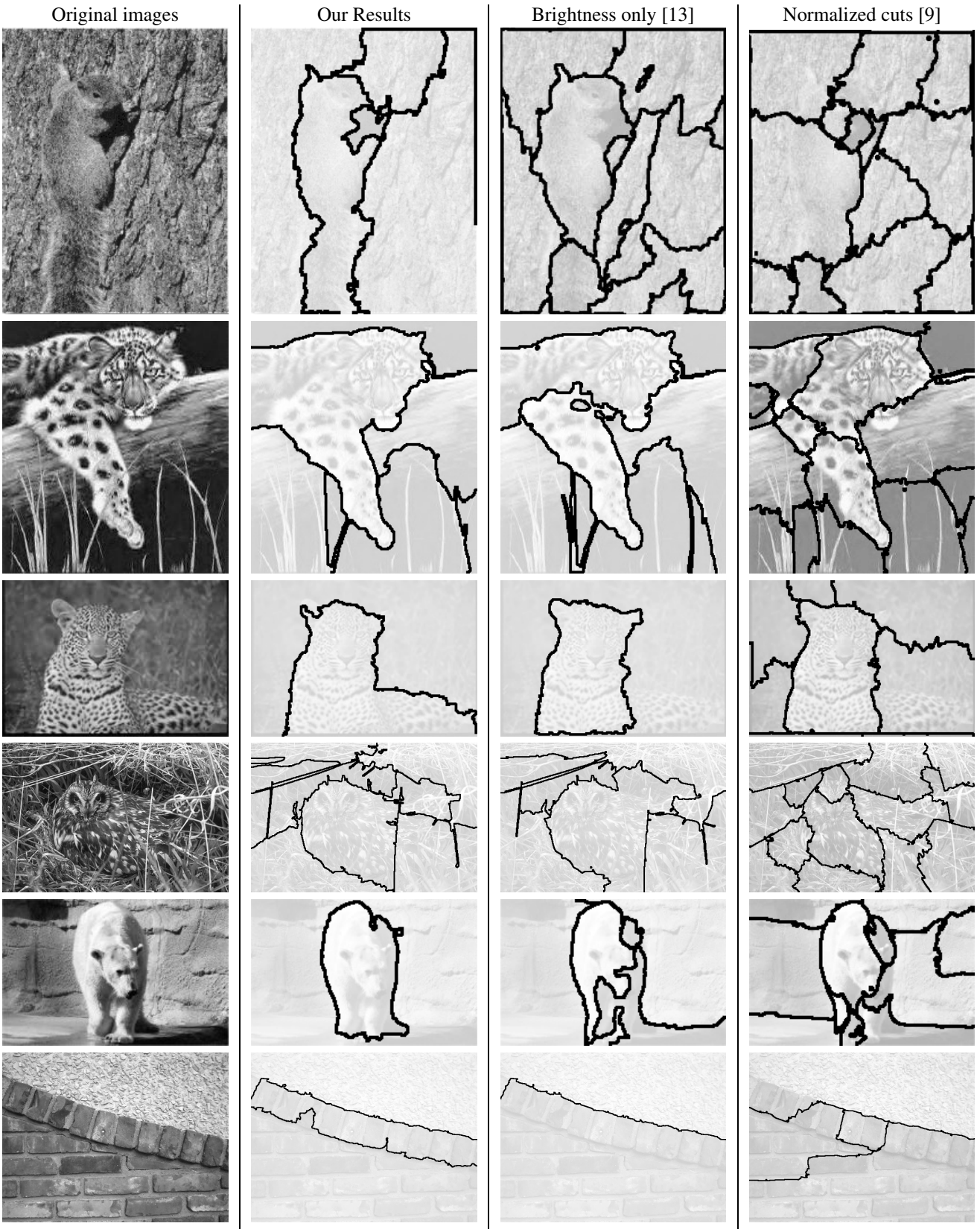


**Figure 1. Experimental results**

Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003) 2-Volume Set
0-7695-1950-4/03 $17.00 © 2003 **IEEE**