Contour Detection and Hierarchical Image Segmentation

Pablo Arbeláez, *Member, IEEE,* Michael Maire, *Member, IEEE,*
Charless Fowlkes, *Member, IEEE,* and Jitendra Malik, *Fellow, IEEE.*

Abstract—This paper investigates two fundamental problems in computer vision: contour detection and image segmentation. We present state-of-the-art algorithms for both of these tasks. Our contour detector combines multiple local cues into a globalization framework based on spectral clustering. Our segmentation algorithm consists of generic machinery for transforming the output of any contour detector into a hierarchical region tree. In this manner, we reduce the problem of image segmentation to that of contour detection. Extensive experimental evaluation demonstrates that both our contour detection and segmentation methods significantly outperform competing algorithms. The automatically generated hierarchical segmentations can be interactively refined by user-specified annotations. Computation at multiple image resolutions provides a means of coupling our system to recognition applications.

1 Introduction

This paper presents a unified approach to contour detection and image segmentation. Contributions include:

- A high performance contour detector, combining local and global image information.
- A method to transform any contour signal into a hierarchy of regions while preserving contour quality.
- Extensive quantitative evaluation and the release of a new annotated dataset.

Figures 1 and 2 summarize our main results. The two Figures represent the evaluation of multiple contour detection (Figure 1) and image segmentation (Figure 2) algorithms on the Berkeley Segmentation Dataset (BSDS300) [1], using the precision-recall framework introduced in [2]. This benchmark operates by comparing machine generated contours to human ground-truth data (Figure 3) and allows evaluation of segmentations in the same framework by regarding region boundaries as contours.

Especially noteworthy in Figure 1 is the contour detector gPb, which compares favorably with other leading techniques, providing equal or better precision for most choices of recall. In Figure 2, gPb-owt-ucm provides universally better performance than alternative segmentation algorithms. We introduced the gPb and gPb-owt-ucm algorithms in [3] and [4], respectively. This paper offers comprehensive versions of these algorithms, motivation behind their design, and additional experiments which support our basic claims.

We begin with a review of the extensive literature on contour detection and image segmentation in Section 2.

- *P. Arbeláez and J. Malik are with the Department of Electrical Engineering and Computer Science, University of California at Berkeley, Berkeley, CA 94720. E-mail: {arbelaez,malik}@eecs.berkeley.edu*
- *M. Maire is with the Department of Electrical Engineering, California Institute of Technology, Pasadena, CA 91125. E-mail: mmaire@caltech.edu*
- *C. Fowlkes is with the Department of Computer Science, University of California at Irvine, Irvine, CA 92697. E-mail: fowlkes@ics.uci.edu*

Section 3 covers the development of the gPb contour detector. We couple multiscale local brightness, color, and texture cues to a powerful globalization framework using spectral clustering. The local cues, computed by applying oriented gradient operators at every location in the image, define an affinity matrix representing the similarity between pixels. From this matrix, we derive a generalized eigenproblem and solve for a fixed number of eigenvectors which encode contour information. Using a classifier to recombine this signal with the local cues, we obtain a large improvement over alternative globalization schemes built on top of similar cues.

To produce high-quality image segmentations, we link this contour detector with a generic grouping algorithm described in Section 4 and consisting of two steps. First, we introduce a new image transformation called the Oriented Watershed Transform for constructing a set of initial regions from an oriented contour signal. Second, using an agglomerative clustering procedure, we form these regions into a hierarchy which can be represented by an Ultrametric Contour Map, the real-valued image obtained by weighting each boundary by its scale of disappearance. We provide experiments on the BSDS300 as well as the BSDS500, a superset newly released here.

Although the precision-recall framework [2] has found widespread use for evaluating contour detectors, considerable effort has also gone into developing metrics to directly measure the quality of regions produced by segmentation algorithms. Noteworthy examples include the Probabilistic Rand Index, introduced in this context by [5], the Variation of Information [6], [7], and the Segmentation Covering criteria used in the PASCAL challenge [8]. We consider all of these metrics and demonstrate that gPb-owt-ucm delivers an across-the-board improvement over existing algorithms.

Sections 5 and 6 explore ways of connecting our purely bottom-up contour and segmentation machinery

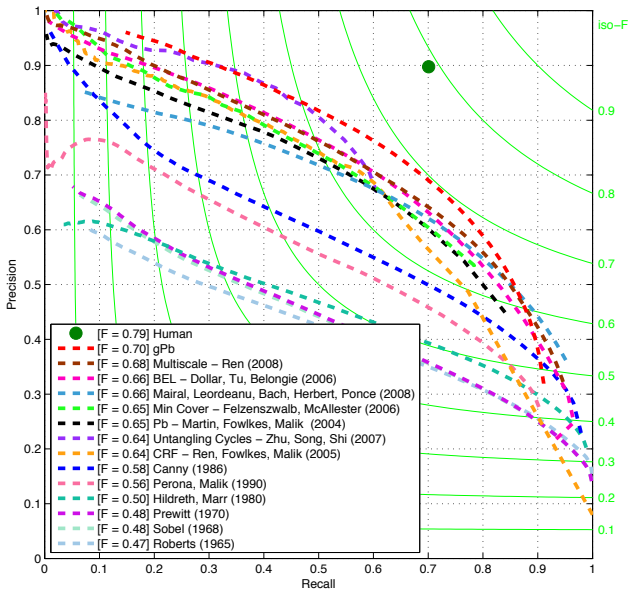

Fig. 1. **Evaluation of contour detectors on the Berkeley Segmentation Dataset (BSDS300) Benchmark** [2]**.** Leading contour detection approaches are ranked according to their maximum F-measure ($\frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$) with respect to human ground-truth boundaries. Iso-F curves are shown in green. Our gPb detector [3] performs significantly better than other algorithms [2], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28] across almost the entire operating regime. Average agreement between human subjects is indicated by the green dot.

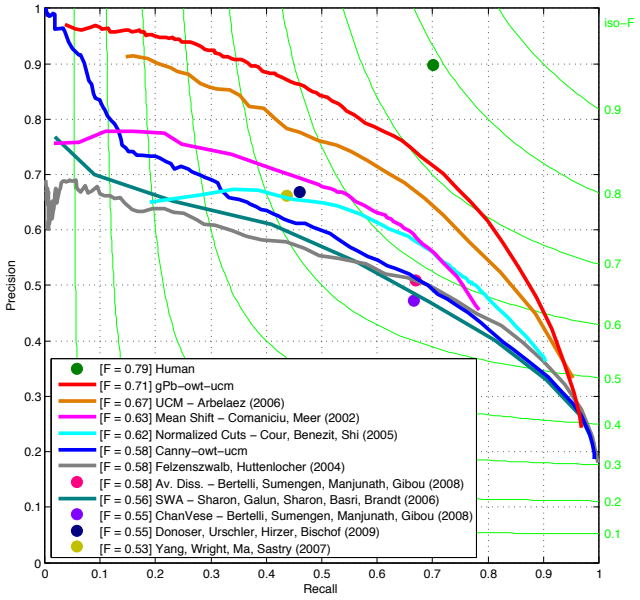

Fig. 2. **Evaluation of segmentation algorithms on the BSDS300 Benchmark.** Paired with our gPb contour detector as input, our hierarchical segmentation algorithm gPb-owt-ucm [4] produces regions whose boundaries match ground-truth better than those produced by other methods [7], [29], [30], [31], [32], [33], [34], [35].

to sources of top-down knowledge. In Section 5, this knowledge source is a human. Our hierarchical region trees serve as a natural starting point for interactive segmentation. With minimal annotation, a user can correct errors in the automatic segmentation and pull out objects of interest from the image. In Section 6, we target top-down object detection algorithms and show how to create multiscale contour and region output tailored to match the scales of interest to the object detector.

Though much remains to be done to take full advantage of segmentation as an intermediate processing layer, recent work has produced payoffs from this endeavor [9], [10], [11], [12], [13]. In particular, our gPb-owt-ucm segmentation algorithm has found use in optical flow [14] and object recognition [15], [16] applications.

2 Previous Work

The problems of contour detection and segmentation are related, but not identical. In general, contour detectors offer no guarantee that they will produce closed contours and hence do not necessarily provide a partition of the image into regions. But, one can always recover closed contours from regions in the form of their boundaries. As an accomplishment here, Section 4 shows how to do the reverse and recover regions from a contour detector.

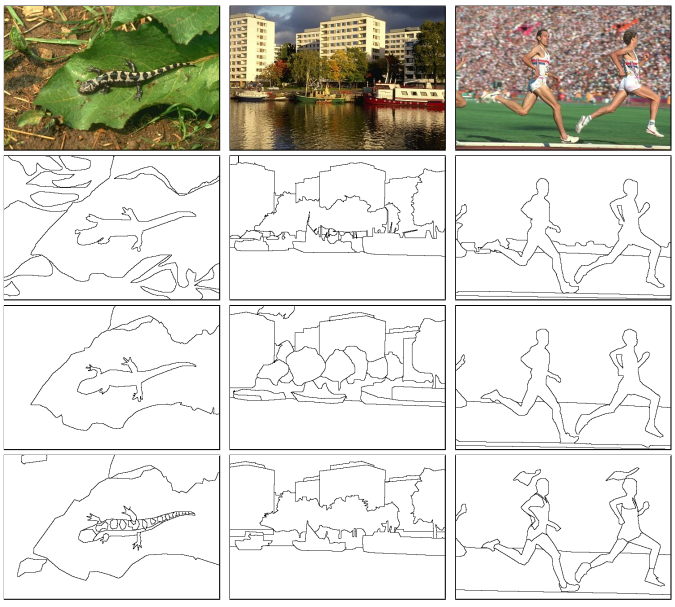

Fig. 3. **Berkeley Segmentation Dataset** [1]**.** ***Top to Bottom:*** Image and ground-truth segment boundaries hand-drawn by three different human subjects. The BSDS300 consists of 200 training and 100 test images, each with multiple ground-truth segmentations. The BSDS500 uses the BSDS300 as training and adds 200 new test images.

Historically, however, there have been different lines of approach to these two problems, which we now review.

2.1 Contours

Early approaches to contour detection aim at quantifying the presence of a boundary at a given image location through local measurements. The Roberts [17], Sobel [18], and Prewitt [19] operators detect edges by convolving a grayscale image with local derivative filters. Marr and Hildreth [20] use zero crossings of the Laplacian of Gaussian operator. The Canny detector [22] also models edges as sharp discontinuities in the brightness channel, adding non-maximum suppression and hysteresis thresholding steps. A richer description can be obtained by considering the response of the image to a family of filters of different scales and orientations. An example is the Oriented Energy approach [21], [36], [37], which uses quadrature pairs of even and odd symmetric filters. Lindeberg [38] proposes a filter-based method with an automatic scale selection mechanism.

More recent local approaches take into account color and texture information and make use of learning techniques for cue combination [2], [26], [27]. Martin *et al.* [2] define gradient operators for brightness, color, and texture channels, and use them as input to a logistic regression classifier for predicting edge strength. Rather than rely on such hand-crafted features, Dollar *et al.* [27] propose a Boosted Edge Learning (BEL) algorithm which attempts to learn an edge classifier in the form of a probabilistic boosting tree [39] from thousands of simple features computed on image patches. An advantage of this approach is that it may be possible to handle cues such as parallelism and completion in the initial classification stage. Mairal *et al.* [26] create both generic and class-specific edge detectors by learning discriminative sparse representations of local image patches. For each class, they learn a discriminative dictionary and use the reconstruction error obtained with each dictionary as feature input to a final classifier.

The large range of scales at which objects may appear in the image remains a concern for these modern local approaches. Ren [28] finds benefit in combining information from multiple scales of the local operators developed by [2]. Additional localization and relative contrast cues, defined in terms of the multiscale detector output, are fed to the boundary classifier. For each scale, the localization cue captures the distance from a pixel to the nearest peak response. The relative contrast cue normalizes each pixel in terms of the local neighborhood.

An orthogonal line of work in contour detection focuses primarily on another level of processing, globalization, that utilizes local detector output. The simplest such algorithms link together high-gradient edge fragments in order to identify extended, smooth contours [40], [41], [42]. More advanced globalization stages are the distinguishing characteristics of several of the recent high-performance methods benchmarked in Figure 1, including our own, which share as a common feature their use of the local edge detection operators of [2].

Ren *et al.* [23] use the Conditional Random Fields (CRF) framework to enforce curvilinear continuity of contours. They compute a constrained Delaunay triangulation (CDT) on top of locally detected contours, yielding a graph consisting of the detected contours along with the new "completion" edges introduced by the triangulation. The CDT is scale-invariant and tends to fill short gaps in the detected contours. By associating a random variable with each contour and each completion edge, they define a CRF with edge potentials in terms of detector response and vertex potentials in terms of junction type and continuation smoothness. They use loopy belief propagation [43] to compute expectations.

Felzenszwalb and McAllester [25] use a different strategy for extracting salient smooth curves from the output of a local contour detector. They consider the set of short oriented line segments that connect pixels in the image to their neighboring pixels. Each such segment is either part of a curve or is a background segment. They assume curves are drawn from a Markov process, the prior distribution on curves favors few per scene, and detector responses are conditionally independent given the labeling of line segments. Finding the optimal line segment labeling then translates into a general weighted min-cover problem in which the elements being covered are the line segments themselves and the objects covering them are drawn from the set of all possible curves and all possible background line segments. Since this problem is NP-hard, an approximate solution is found using a greedy "cost per pixel" heuristic.

Zhu *et al.* [24] also start with the output of [2] and create a weighted edgel graph, where the weights measure directed collinearity between neighboring edgels. They propose detecting closed topological cycles in this graph by considering the complex eigenvectors of the normalized random walk matrix. This procedure extracts both closed contours and smooth curves, as edgel chains are allowed to loop back at their termination points.

2.2 Regions

A broad family of approaches to segmentation involve integrating features such as brightness, color, or texture over local image patches and then clustering those features based on, *e.g.*, fitting mixture models [7], [44], mode-finding [34], or graph partitioning [32], [45], [46], [47]. Three algorithms in this category appear to be the most widely used as sources of image segments in recent applications, due to a combination of reasonable performance and publicly available implementations.

The graph based region merging algorithm advocated by Felzenszwalb and Huttenlocher (Felz-Hutt) [32] attempts to partition image pixels into components such that the resulting segmentation is neither too coarse nor too fine. Given a graph in which pixels are nodes and edge weights measure the dissimilarity between nodes (*e.g.* color differences), each node is initially placed in its own component. Define the internal difference of a component $Int(R)$ as the largest weight in the minimum

spanning tree of R. Considering edges in non-decreasing order by weight, each step of the algorithm merges components R_1 and R_2 connected by the current edge if the edge weight is less than:

$$min(Int(R_1) + \tau(R_1), Int(R_2) + \tau(R_2)) \quad (1)$$

where $\tau(R) = k/|R|$. k is a scale parameter that can be used to set a preference for component size.

The Mean Shift algorithm [34] offers an alternative clustering framework. Here, pixels are represented in the joint spatial-range domain by concatenating their spatial coordinates and color values into a single vector. Applying mean shift filtering in this domain yields a convergence point for each pixel. Regions are formed by grouping together all pixels whose convergence points are closer than h_s in the spatial domain and h_r in the range domain, where h_s and h_r are respective bandwidth parameters. Additional merging can also be performed to enforce a constraint on minimum region area.

Spectral graph theory [48], and in particular the Normalized Cuts criterion [45], [46], provides a way of integrating global image information into the grouping process. In this framework, given an affinity matrix W whose entries encode the similarity between pixels, one defines diagonal matrix $D_{ii} = \sum_j W_{ij}$ and solves for the generalized eigenvectors of the linear system:

$$(D - W)\mathbf{v} = \lambda D\mathbf{v} \quad (2)$$

Traditionally, after this step, K-means clustering is applied to obtain a segmentation into regions. This approach often breaks uniform regions where the eigenvectors have smooth gradients. One solution is to reweight the affinity matrix [47]; others have proposed alternative graph partitioning formulations [49], [50], [51].

A recent variant of Normalized Cuts for image segmentation is the Multiscale Normalized Cuts (NCuts) approach of Cour *et al.* [33]. The fact that W must be sparse, in order to avoid a prohibitively expensive computation, limits the naive implementation to using only local pixel affinities. Cour *et al.* solve this limitation by computing sparse affinity matrices at multiple scales, setting up cross-scale constraints, and deriving a new eigenproblem for this constrained multiscale cut.

Sharon *et al.* [31] propose an alternative to improve the computational efficiency of Normalized Cuts. This approach, inspired by algebraic multigrid, iteratively coarsens the original graph by selecting a subset of nodes such that each variable on the fine level is strongly coupled to one on the coarse level. The same merging strategy is adopted in [52], where the strong coupling of a subset S of the graph nodes V is formalized as:

$$\frac{\sum_{j \in S} p_{ij}}{\sum_{j \in V} p_{ij}} > \psi \qquad \forall i \in V - S \quad (3)$$

where ψ is a constant and p_{ij} the probability of merging i and j, estimated from brightness and texture similarity.

Many approaches to image segmentation fall into a different category than those covered so far, relying on the formulation of the problem in a variational framework. An example is the model proposed by Mumford and Shah [53], where the segmentation of an observed image u_0 is given by the minimization of the functional:

$$\mathcal{F}(u, C) = \int_\Omega (u - u_0)^2 dx + \mu \int_{\Omega \setminus C} |\nabla(u)|^2 dx + \nu |C| \quad (4)$$

where u is piecewise smooth in $\Omega \setminus C$ and μ, ν are weighting parameters. Theoretical properties of this model can be found in, *e.g.* [53], [54]. Several algorithms have been developed to minimize the energy (4) or its simplified version, where u is piecewise constant in $\Omega \setminus C$. Koepfler *et al.* [55] proposed a region merging method for this purpose. Chan and Vese [56], [57] follow a different approach, expressing (4) in the level set formalism of Osher and Sethian [58], [59]. Bertelli *et al.* [30] extend this approach to more general cost functions based on pairwise pixel similarities. Recently, Pock *et al.* [60] proposed to solve a convex relaxation of (4), thus obtaining robustness to initialization. Donoser *et al.* [29] subdivide the problem into several figure/ground segmentations, each initialized using low-level saliency and solved by minimizing an energy based on Total Variation.

2.3 Benchmarks

Though much of the extensive literature on contour detection predates its development, the BSDS [2] has since found wide acceptance as a benchmark for this task [23], [24], [25], [26], [27], [28], [35], [61]. The standard for evaluating segmentations algorithms is less clear.

One option is to regard the segment boundaries as contours and evaluate them as such. However, a methodology that directly measures the quality of the segments is also desirable. Some types of errors, *e.g.* a missing pixel in the boundary between two regions, may not be reflected in the boundary benchmark, but can have substantial consequences for segmentation quality, *e.g.* incorrectly merging large regions. One might argue that the boundary benchmark favors contour detectors over segmentation methods, since the former are not burdened with the constraint of producing closed curves. We therefore also consider various region-based metrics.

2.3.1 Variation of Information

The Variation of Information metric was introduced for the purpose of clustering comparison [6]. It measures the distance between two segmentations in terms of their average conditional entropy given by:

$$VI(S, S') = H(S) + H(S') - 2I(S, S') \quad (5)$$

where H and I represent respectively the entropies and mutual information between two clusterings of data S and S'. In our case, these clusterings are test and ground-truth segmentations. Although VI possesses some interesting theoretical properties [6], its perceptual meaning and applicability in the presence of several ground-truth segmentations remains unclear.

2.3.2 Rand Index

Originally, the Rand Index [62] was introduced for general clustering evaluation. It operates by comparing the compatibility of assignments between pairs of elements in the clusters. The Rand Index between test and ground-truth segmentations S and G is given by the sum of the number of pairs of pixels that have the same label in S and G and those that have different labels in both segmentations, divided by the total number of pairs of pixels. Variants of the Rand Index have been proposed [5], [7] for dealing with the case of multiple ground-truth segmentations. Given a set of ground-truth segmentations $\{G_k\}$, the Probabilistic Rand Index is defined as:

$$PRI(S, \{G_k\}) = \frac{1}{T} \sum_{i<j} [c_{ij} p_{ij} + (1 - c_{ij})(1 - p_{ij})] \quad (6)$$

where c_{ij} is the event that pixels i and j have the same label and p_{ij} its probability. T is the total number of pixel pairs. Using the sample mean to estimate p_{ij}, (6) amounts to averaging the Rand Index among different ground-truth segmentations. The PRI has been reported to suffer from a small dynamic range [5], [7], and its values across images and algorithms are often similar. In [5], this drawback is addressed by normalization with an empirical estimation of its expected value.

2.3.3 Segmentation Covering

The *overlap* between two regions R and R', defined as:

$$\mathcal{O}(R, R') = \frac{|R \cap R'|}{|R \cup R'|} \quad (7)$$

has been used for the evaluation of the pixel-wise classification task in recognition [8], [11]. We define the *covering* of a segmentation S by a segmentation S' as:

$$\mathcal{C}(S' \to S) = \frac{1}{N} \sum_{R \in S} |R| \cdot \max_{R' \in S'} \mathcal{O}(R, R') \quad (8)$$

where N denotes the total number of pixels in the image.

Similarly, the covering of a machine segmentation S by a family of ground-truth segmentations $\{G_i\}$ is defined by first covering S separately with each human segmentation G_i, and then averaging over the different humans. To achieve perfect covering the machine segmentation must explain all of the human data. We can then define two quality descriptors for regions: the covering of S by $\{G_i\}$ and the covering of $\{G_i\}$ by S.

3 Contour Detection

As a starting point for contour detection, we consider the work of Martin *et al.* [2], who define a function $Pb(x, y, \theta)$ that predicts the posterior probability of a boundary with orientation θ at each image pixel (x, y) by measuring the difference in local image brightness, color, and texture channels. In this section, we review these cues, introduce our own multiscale version of the Pb detector, and describe the new globalization method we run on top of this multiscale local detector.

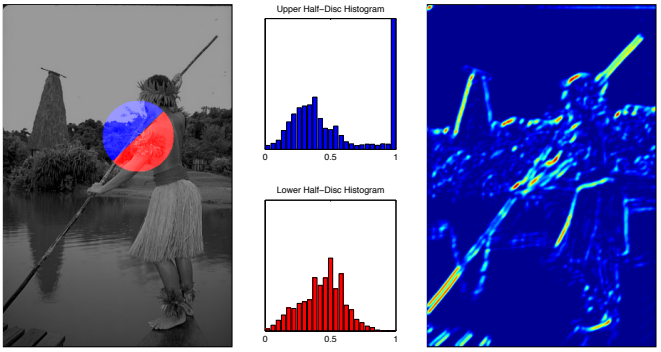

Fig. 4. **Oriented gradient of histograms.** Given an intensity image, consider a circular disc centered at each pixel and split by a diameter at angle θ. We compute histograms of intensity values in each half-disc and output the χ^2 distance between them as the gradient magnitude. The blue and red distributions shown in the middle panel are the histograms of the pixel brightness values in the blue and red regions, respectively, in the left image. The right panel shows an example result for a disc of radius 5 pixels at orientation $\theta = \frac{\pi}{4}$ after applying a second-order Savitzky-Golay smoothing filter to the raw histogram difference output. Note that the left panel displays a larger disc (radius 50 pixels) for illustrative purposes.

3.1 Brightness, Color, Texture Gradients

The basic building block of the Pb contour detector is the computation of an oriented gradient signal $G(x, y, \theta)$ from an intensity image I. This computation proceeds by placing a circular disc at location (x, y) split into two half-discs by a diameter at angle θ. For each half-disc, we histogram the intensity values of the pixels of I covered by it. The gradient magnitude G at location (x, y) is defined by the χ^2 distance between the two half-disc histograms g and h:

$$\chi^2(g, h) = \frac{1}{2} \sum_i \frac{(g(i) - h(i))^2}{g(i) + h(i)} \quad (9)$$

We then apply second-order Savitzky-Golay filtering [63] to enhance local maxima and smooth out multiple detection peaks in the direction orthogonal to θ. This is equivalent to fitting a cylindrical parabola, whose axis is orientated along direction θ, to a local 2D window surrounding each pixel and replacing the response at the pixel with that estimated by the fit.

Figure 4 shows an example. This computation is motivated by the intuition that contours correspond to image discontinuities and histograms provide a robust mechanism for modeling the content of an image region. A strong oriented gradient response means a pixel is likely to lie on the boundary between two distinct regions.

The Pb detector combines the oriented gradient signals obtained from transforming an input image into four separate feature channels and processing each channel independently. The first three correspond to the channels of the CIE Lab colorspace, which we refer to

Fig. 5. **Filters for creating textons.** We use 8 oriented even- and odd-symmetric Gaussian derivative filters and a center-surround (difference of Gaussians) filter.

as the brightness, color a, and color b channels. For grayscale images, the brightness channel is the image itself and no color channels are used.

The fourth channel is a texture channel, which assigns each pixel a texton id. These assignments are computed by another filtering stage which occurs prior to the computation of the oriented gradient of histograms. This stage converts the input image to grayscale and convolves it with the set of 17 Gaussian derivative and center-surround filters shown in Figure 5. Each pixel is associated with a (17-dimensional) vector of responses, containing one entry for each filter. These vectors are then clustered using K-means. The cluster centers define a set of image-specific textons and each pixel is assigned the integer id in $[1, K]$ of the closest cluster center. Experiments show choosing $K = 32$ textons to be sufficient.

We next form an image where each pixel has an integer value in $[1, K]$, as determined by its texton id. An example can be seen in Figure 6 (left column, fourth panel from top). On this image, we compute differences of histograms in oriented half-discs in the same manner as for the brightness and color channels.

Obtaining $G(x, y, \theta)$ for arbitrary input I is thus the core operation on which our local cues depend. In the appendix, we provide a novel approximation scheme for reducing the complexity of this computation.

3.2 Multiscale Cue Combination

We now introduce our own multiscale extension of the Pb detector reviewed above. Note that Ren [28] introduces a different, more complicated, and similarly performing multiscale extension in work contemporaneous with our own [3], and also suggests possible reasons Martin *et al.* [2] did not see performance improvements in their original multiscale experiments, including their use of smaller images and their choice of scales.

In order to detect fine as well as coarse structures, we consider gradients at three scales: $[\frac{\sigma}{2}, \sigma, 2\sigma]$ for each of the brightness, color, and texture channels. Figure 6 shows an example of the oriented gradients obtained for each channel. For the brightness channel, we use $\sigma = 5$ pixels, while for color and texture we use $\sigma = 10$ pixels. We then linearly combine these local cues into a single multiscale oriented signal:

$$mPb(x, y, \theta) = \sum_s \sum_i \alpha_{i,s} G_{i,\sigma(i,s)}(x, y, \theta) \qquad (10)$$

where s indexes scales, i indexes feature channels (brightness, color a, color b, texture), and $G_{i,\sigma(i,s)}(x, y, \theta)$ measures the histogram difference in channel i between

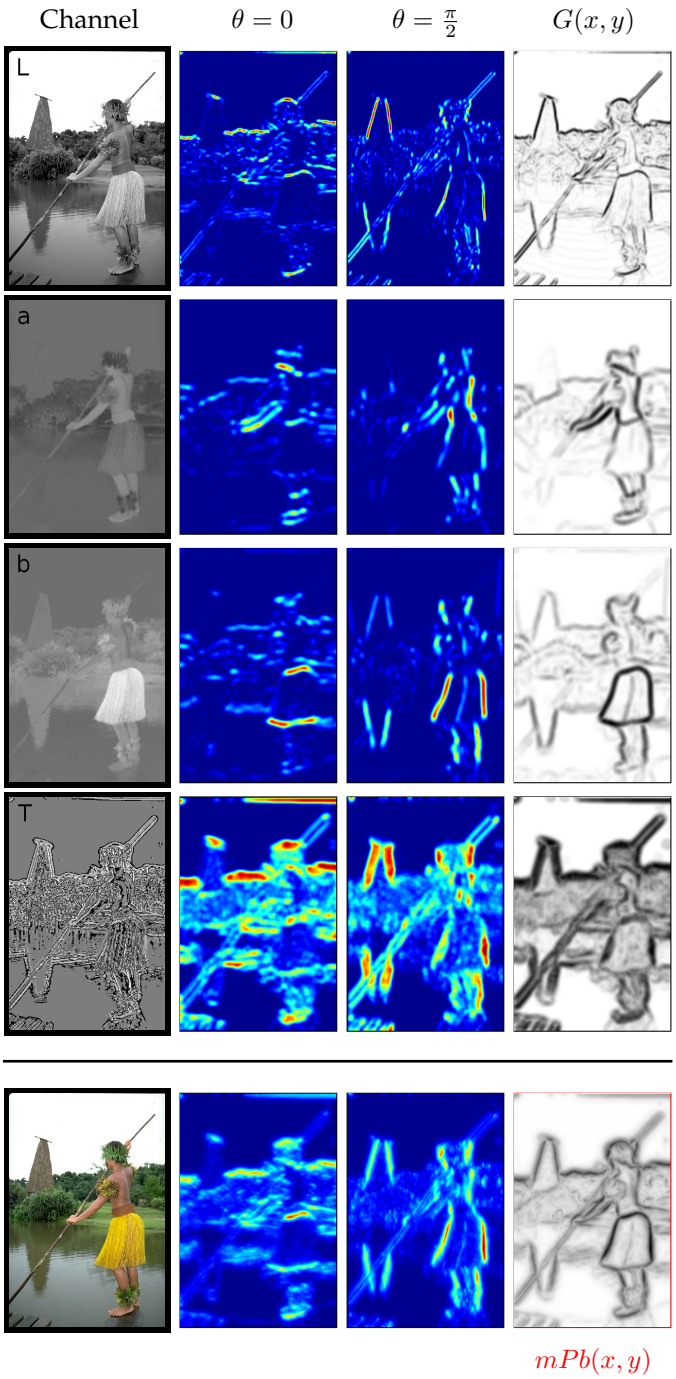

Fig. 6. **Multiscale Pb.** ***Left Column, Top to Bottom:*** The brightness and color a and b channels of Lab color space, and the texton channel computed using image-specific textons, followed by the input image. ***Rows:*** Next to each channel, we display the oriented gradient of histograms (as outlined in Figure 4) for $\theta = 0$ and $\theta = \frac{\pi}{2}$ (horizontal and vertical), and the maximum response over eight orientations in $[0, \pi)$ (right column). Beside the original image, we display the combination of oriented gradients across all four channels and across three scales. The lower right panel (outlined in red) shows mPb, the final output of the multiscale contour detector.

Fig. 7. **Spectral Pb.** ***Left:*** Image. ***Middle Left:*** The thinned non-max suppressed multiscale Pb signal defines a sparse affinity matrix connecting pixels within a fixed radius. Pixels i and j have a low affinity as a strong boundary separates them, whereas i and k have high affinity. ***Middle:*** First four generalized eigenvectors resulting from spectral clustering. ***Middle Right:*** Partitioning the image by running K-means clustering on the eigenvectors erroneously breaks smooth regions. ***Right:*** Instead, we compute gradients of the eigenvectors, transforming them back into a contour signal.

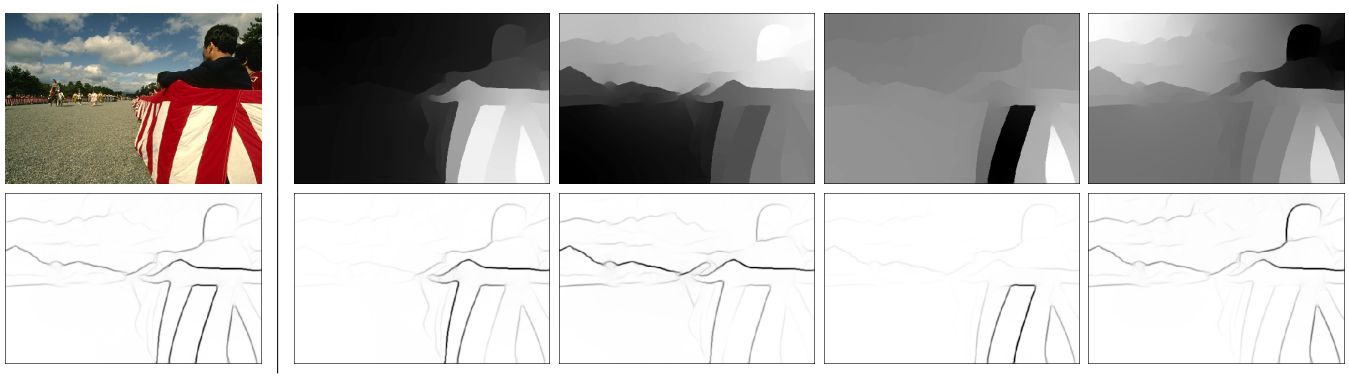

Fig. 8. **Eigenvectors carry contour information.** ***Left:*** Image and maximum response of spectral Pb over orientations, $sPb(x,y) = \max_\theta\{sPb(x,y,\theta)\}$. ***Right Top:*** First four generalized eigenvectors, $\mathbf{v_1}, ..., \mathbf{v_4}$, used in creating sPb. ***Right Bottom:*** Maximum gradient response over orientations, $\max_\theta\{\nabla_\theta \mathbf{v_k}(x,y)\}$, for each eigenvector.

two halves of a disc of radius $\sigma(i,s)$ centered at (x,y) and divided by a diameter at angle θ. The parameters $\alpha_{i,s}$ weight the relative contribution of each gradient signal. In our experiments, we sample θ at eight equally spaced orientations in the interval $[0,\pi)$. Taking the maximum response over orientations yields a measure of boundary strength at each pixel:

$$mPb(x,y) = \max_\theta\{mPb(x,y,\theta)\} \tag{11}$$

An optional non-maximum suppression step [22] produces thinned, real-valued contours.

In contrast to [2] and [28] which use a logistic regression classifier to combine cues, we learn the weights $\alpha_{i,s}$ by gradient ascent on the F-measure using the training images and corresponding ground-truth of the BSDS.

3.3 Globalization

Spectral clustering lies at the heart of our globalization machinery. The key element differentiating the algorithm described in this section from other approaches [45], [47] is the "soft" manner in which we use the eigenvectors obtained from spectral partitioning.

As input to the spectral clustering stage, we construct a sparse symmetric affinity matrix W using the *intervening contour* cue [49], [64], [65], the maximal value of mPb along a line connecting two pixels. We connect all pixels i and j within a fixed radius r with affinity:

$$W_{ij} = \exp\left(-\max_{p\in\overline{ij}}\{mPb(p)\}/\rho\right) \tag{12}$$

where $\overline{ij}$ is the line segment connecting i and j and ρ is a constant. We set $r = 5$ pixels and $\rho = 0.1$.

In order to introduce global information, we define $D_{ii} = \sum_j W_{ij}$ and solve for the generalized eigenvectors $\{\mathbf{v_0}, \mathbf{v_1}, ..., \mathbf{v_n}\}$ of the system $(D - W)\mathbf{v} = \lambda D\mathbf{v}$ (2), corresponding to the $n+1$ smallest eigenvalues $0 = \lambda_0 \leq \lambda_1 \leq ... \leq \lambda_n$. Figure 7 displays an example with four eigenvectors. In practice, we use $n = 16$.

At this point, the standard Normalized Cuts approach associates with each pixel a length n descriptor formed from entries of the n eigenvectors and uses a clustering

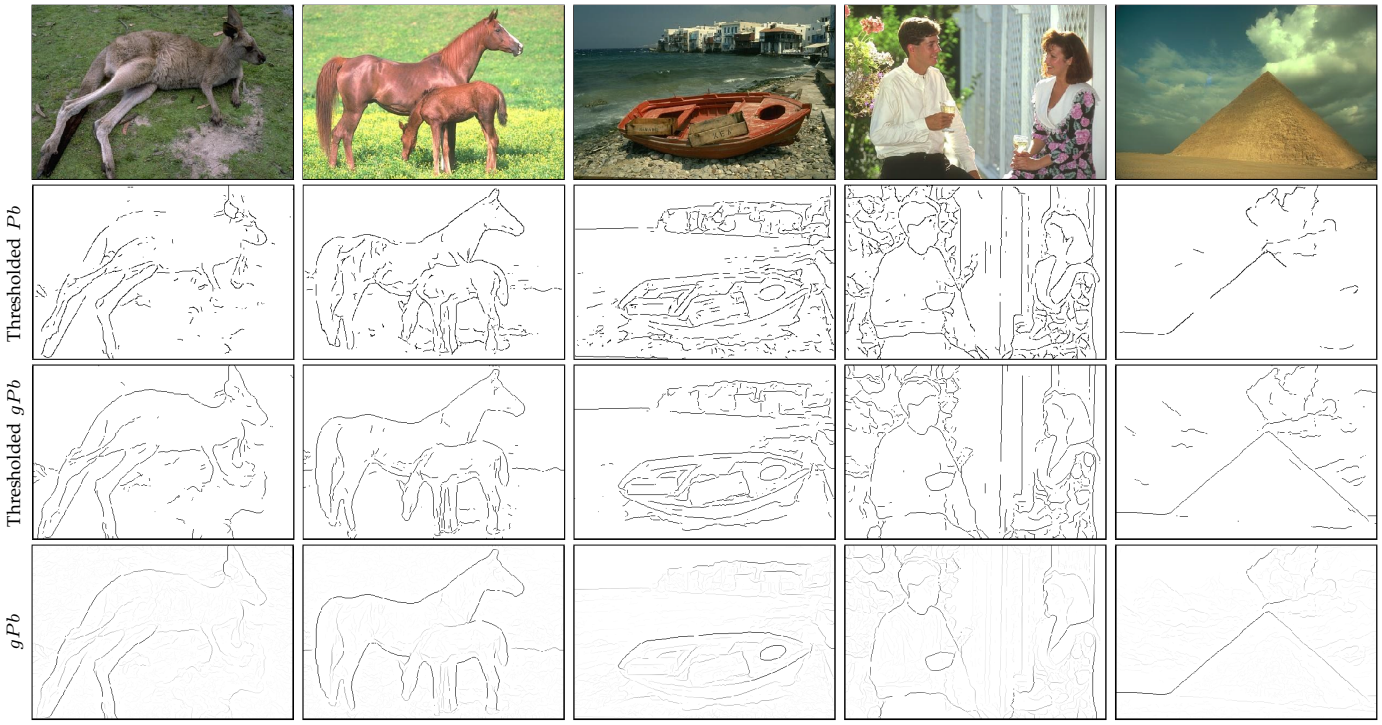

Fig. 9. **Benefits of globalization.** When compared with the local detector Pb, our detector gPb reduces clutter and completes contours. The thresholds shown correspond to the points of maximal F-measure on the curves in Figure 1.

algorithm such as K-means to create a hard partition of the image. Unfortunately, this can lead to an incorrect segmentation as large uniform regions in which the eigenvectors vary smoothly are broken up. Figure 7 shows an example for which such gradual variation in the eigenvectors across the sky region results in an incorrect partition.

To circumvent this difficulty, we observe that the eigenvectors themselves carry contour information. Treating each eigenvector $\mathbf{v_k}$ as an image, we convolve with Gaussian directional derivative filters at multiple orientations θ, obtaining oriented signals $\{\nabla_\theta \mathbf{v_k}(x,y)\}$. Taking derivatives in this manner ignores the smooth variations that previously lead to errors. The information from different eigenvectors is then combined to provide the "spectral" component of our boundary detector:

$$sPb(x,y,\theta) = \sum_{k=1}^{n} \frac{1}{\sqrt{\lambda_k}} \cdot \nabla_\theta \mathbf{v_k}(x,y) \qquad (13)$$

where the weighting by $1/\sqrt{\lambda_k}$ is motivated by the physical interpretation of the generalized eigenvalue problem as a mass-spring system [66]. Figures 7 and 8 present examples of the eigenvectors, their directional derivatives, and the resulting sPb signal.

The signals mPb and sPb convey different information, as the former fires at all the edges while the latter extracts only the most salient curves in the image. We found that a simple linear combination is enough to benefit from both behaviors. Our final *globalized probability of boundary* is then written as a weighted sum of local

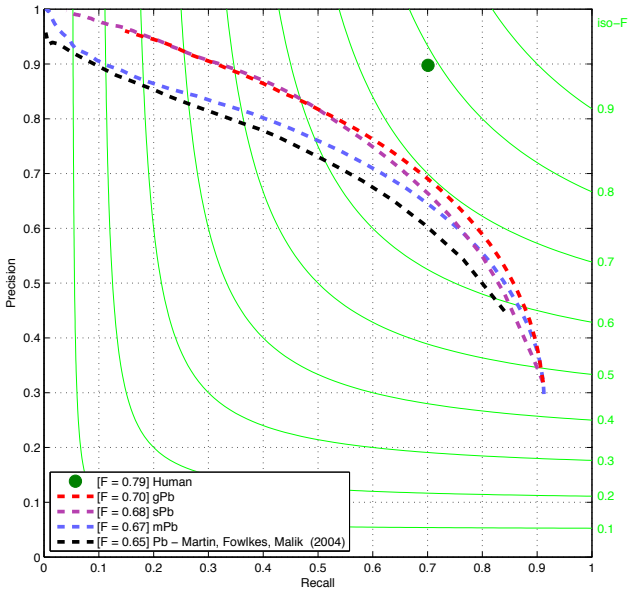

Fig. 10. **Globalization improves contour detection.** The spectral Pb detector (sPb), derived from the eigenvectors of a spectral partitioning algorithm, improves the precision of the local multiscale Pb signal (mPb) used as input. Global Pb (gPb), a learned combination of the two, provides uniformly better performance. Also note the benefit of using multiple scales (mPb) over single scale Pb. Results shown on the BSDS300.

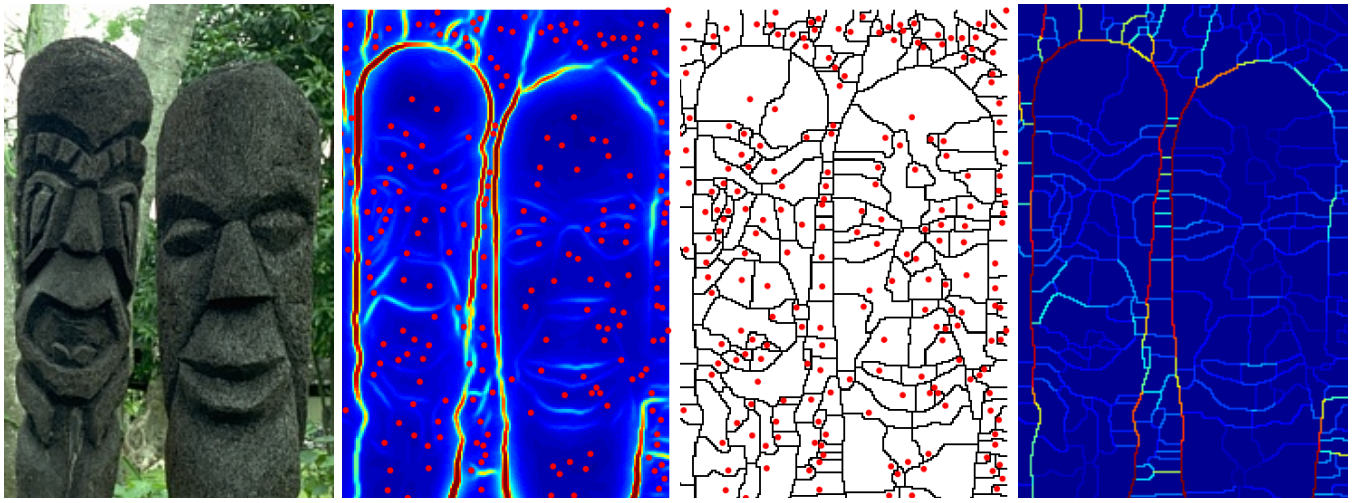

Fig. 11. **Watershed Transform.** ***Left:*** Image. ***Middle Left:*** Boundary strength $E(x, y)$. We regard $E(x, y)$ as a topographic surface and flood it from its local minima. ***Middle Right:*** This process partitions the image into catchment basins $\mathcal{P}_0$ and arcs $\mathcal{K}_0$. There is exactly one basin per local minimum and the arcs coincide with the locations where the floods originating from distinct minima meet. Local minima are marked with red dots. ***Right:*** Each arc weighted by the mean value of $E(x, y)$ along it. This weighting scheme produces artifacts, such as the strong horizontal contours in the small gap between the two statues.

and spectral signals:

$$gPb(x, y, \theta) = \sum_s \sum_i \beta_{i,s} G_{i,\sigma(i,s)}(x, y, \theta) + \gamma \cdot sPb(x, y, \theta) \tag{14}$$

We subsequently rescale gPb using a sigmoid to match a probabilistic interpretation. As with mPb (10), the weights $\beta_{i,s}$ and γ are learned by gradient ascent on the F-measure using the BSDS training images.

3.4 Results

Qualitatively, the combination of the multiscale cues with our globalization machinery translates into a reduction of clutter edges and completion of contours in the output, as shown in Figure 9.

Figure 10 breaks down the contributions of the multiscale and spectral signals to the performance of gPb. These precision-recall curves show that the reduction of false positives due to the use of global information in sPb is concentrated in the high thresholds, while gPb takes the best of both worlds, relying on sPb in the high precision regime and on mPb in the high recall regime.

Looking again at the comparison of contour detectors on the BSDS300 benchmark in Figure 1, the mean improvement in precision of gPb with respect to the single scale Pb is 10% in the recall range $[0.1, 0.9]$.

4 Segmentation

The nonmax suppressed gPb contours produced in the previous section are often not closed and hence do not partition the image into regions. These contours may still be useful, *e.g.* as a signal on which to compute image descriptors. However, closed regions offer additional advantages. Regions come with their own scale estimates and provide natural domains for computing features used in recognition. Many visual tasks can also benefit from the complexity reduction achieved by transforming an image with millions of pixels into a few hundred or thousand "superpixels" [67].

In this section, we show how to recover closed contours, while preserving the gains in boundary quality achieved in the previous section. Our algorithm, first reported in [4], builds a hierarchical segmentation by exploiting the information in the contour signal. We introduce a new variant of the watershed transform [68], [69], the Oriented Watershed Transform (OWT), for producing a set of initial regions from contour detector output. We then construct an Ultrametric Contour Map (UCM) [35] from the boundaries of these initial regions.

This sequence of operations (OWT-UCM) can be seen as generic machinery for going from contours to a hierarchical region tree. Contours encoded in the resulting hierarchical segmentation retain real-valued weights indicating their likelihood of being a true boundary. For a given threshold, the output is a set of closed contours that can be treated as either a segmentation or as a boundary detector for the purposes of benchmarking.

To describe our algorithm in the most general setting, we now consider an arbitrary contour detector, whose output $E(x, y, \theta)$ predicts the probability of an image boundary at location (x, y) and orientation θ.

4.1 Oriented Watershed Transform

Using the contour signal, we first construct a finest partition for the hierarchy, an over-segmentation whose regions determine the highest level of detail considered.

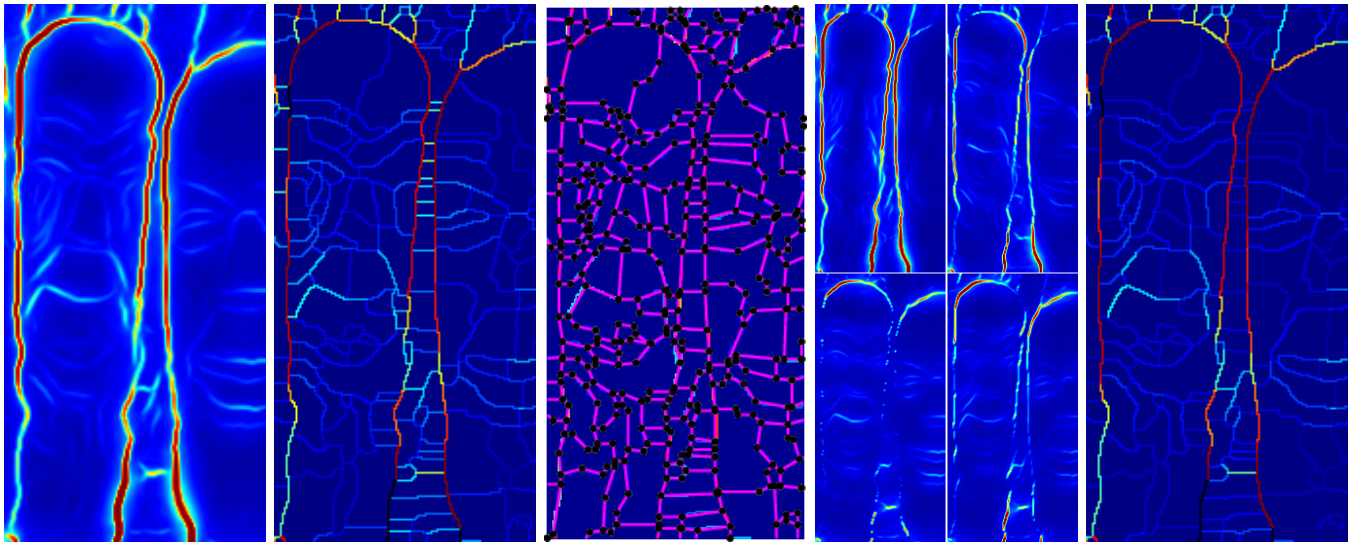

Fig. 12. **Oriented Watershed Transform.** ***Left:*** Input boundary signal $E(x,y) = \max_\theta E(x,y,\theta)$. ***Middle Left:*** Watershed arcs computed from $E(x,y)$. Note that thin regions give rise to artifacts. ***Middle:*** Watershed arcs with an approximating straight line segment subdivision overlaid. We compute this subdivision in a scale-invariant manner by recursively breaking an arc at the point maximally distant from the straight line segment connecting its endpoints, as shown in Figure 13. Subdivision terminates when the distance from the line segment to every point on the arc is less than a fixed fraction of the segment length. ***Middle Right:*** Oriented boundary strength $E(x,y,\theta)$ for four orientations θ. In practice, we sample eight orientations. ***Right:*** Watershed arcs reweighted according to E at the orientation of their associated line segments. Artifacts, such as the horizontal contours breaking the long skinny regions, are suppressed as their orientations do not agree with the underlying $E(x,y,\theta)$ signal.

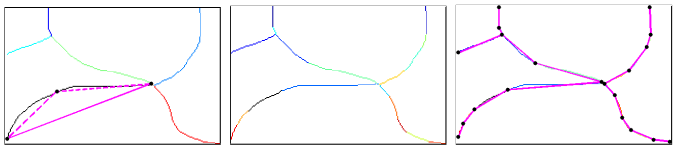

Fig. 13. **Contour subdivision.** ***Left:*** Initial arcs color-coded. If the distance from any point on an arc to the straight line segment connecting its endpoints is greater than a fixed fraction of the segment length, we subdivide the arc at the maximally distant point. An example is shown for one arc, with the dashed segments indicating the new subdivision. ***Middle:*** The final set of arcs resulting from recursive application of the scale-invariant subdivision procedure. ***Right:*** Approximating straight line segments overlaid on the subdivided arcs.

This is done by computing $E(x,y) = \max_\theta E(x,y,\theta)$, the maximal response of the contour detector over orientations. We take the regional minima of $E(x,y)$ as seed locations for homogeneous segments and apply the watershed transform used in mathematical morphology [68], [69] on the topographic surface defined by $E(x,y)$. The catchment basins of the minima, denoted $\mathcal{P}_0$, provide the regions of the finest partition and the corresponding watershed arcs, $\mathcal{K}_0$, the possible locations of the boundaries.

Figure 11 shows an example of the standard watershed transform. Unfortunately, simply weighting each arc by the mean value of $E(x,y)$ for the pixels on the arc can introduce artifacts. The root cause of this problem is the fact that the contour detector produces a spatially extended response around strong boundaries. For example, a pixel could lie near but not on a strong vertical contour. If this pixel also happens to belong to a horizontal watershed arc, that arc would be erroneously upweighted. Several such cases can be seen in Figure 11. As we flood from all local minima, the initial watershed oversegmentation contains many arcs that should be weak, yet intersect nearby strong boundaries.

To correct this problem, we enforce consistency between the strength of the boundaries of $\mathcal{K}_0$ and the underlying $E(x,y,\theta)$ signal in a modified procedure, which we call the Oriented Watershed Transform (OWT), illustrated in Figure 12. As the first step in this reweighting process, we estimate an orientation at each pixel on an arc from the local geometry of the arc itself. These orientations are obtained by approximating the watershed arcs with line segments as shown in Figure 13. We recursively subdivide any arc which is not well fit by the line segment connecting its endpoints. By expressing the approximation criterion in terms of the maximum distance of a point on the arc from the line segment as a fraction of the line segment length, we obtain a scale-invariant subdivision. We assign each pixel (x,y) on a subdivided arc the orientation $o(x,y) \in [0,\pi)$ of the corresponding line segment.

Next, we use the oriented contour detector output

Fig. 14. **Hierarchical segmentation from contours.** ***Far Left:*** Image. ***Left:*** Maximal response of contour detector gPb over orientations. ***Middle Left:*** Weighted contours resulting from the Oriented Watershed Transform - Ultrametric Contour Map (OWT-UCM) algorithm using gPb as input. This single weighted image encodes the entire hierarchical segmentation. By construction, applying any threshold to it is guaranteed to yield a set of closed contours (the ones with weights above the threshold), which in turn define a segmentation. Moreover, the segmentations are nested. Increasing the threshold is equivalent to removing contours and merging the regions they separated. ***Middle Right:*** The initial oversegmentation corresponding to the finest level of the UCM, with regions represented by their mean color. ***Right and Far Right:*** Contours and corresponding segmentation obtained by thresholding the UCM at level 0.5.

$E(x, y, \theta)$, to assign each arc pixel (x, y) a boundary strength of $E(x, y, o(x, y))$. We quantize $o(x, y)$ in the same manner as θ, so this operation is a simple lookup. Finally, each original arc in $\mathcal{K}_0$ is assigned weight equal to average boundary strength of the pixels it contains. Comparing the middle left and far right panels of Figure 12 shows this reweighting scheme removes artifacts.

4.2 Ultrametric Contour Map

Contours have the advantage that it is fairly straightforward to represent uncertainty in the presence of a true underlying contour, *i.e.* by associating a binary random variable to it. One can interpret the boundary strength assigned to an arc by the Oriented Watershed Transform (OWT) of the previous section as an estimate of the probability of that arc being a true contour.

It is not immediately obvious how to represent uncertainty about a segmentation. One possibility, which we exploit here, is the Ultrametric Contour Map (UCM) [35] which defines a duality between closed, non-self-intersecting weighted contours and a hierarchy of regions. The base level of this hierarchy respects even weak contours and is thus an oversegmentation of the image. Upper levels of the hierarchy respect only strong contours, resulting in an under-segmentation. Moving between levels offers a continuous trade-off between these extremes. This shift in representation from a single segmentation to a nested collection of segmentations frees later processing stages to use information from multiple levels or select a level based on additional knowledge.

Our hierarchy is constructed by a greedy graph-based region merging algorithm. We define an initial graph $G = (\mathcal{P}_0, \mathcal{K}_0, W(\mathcal{K}_0))$, where the nodes are the regions $\mathcal{P}_0$, the links are the arcs $\mathcal{K}_0$ separating adjacent regions, and the weights $W(\mathcal{K}_0)$ are a measure of dissimilarity between regions. The algorithm proceeds by sorting the links by similarity and iteratively merging the most similar regions. Specifically:

1) Select minimum weight contour:
   $C^* = \text{argmin}_{C \in \mathcal{K}_0} W(C)$.
2) Let $R_1, R_2 \in \mathcal{P}_0$ be the regions separated by C^*.
3) Set $R = R_1 \cup R_2$, and update:
   $\mathcal{P}_0 \leftarrow \mathcal{P}_0 \backslash \{R_1, R_2\} \cup \{R\}$ and $\mathcal{K}_0 \leftarrow \mathcal{K}_0 \backslash \{C^*\}$.
4) Stop if $\mathcal{K}_0$ is empty.
   Otherwise, update weights $W(\mathcal{K}_0)$ and repeat.

This process produces a tree of regions, where the leaves are the initial elements of $\mathcal{P}_0$, the root is the entire image, and the regions are ordered by the inclusion relation.

We define dissimilarity between two adjacent regions as the average strength of their common boundary in $\mathcal{K}_0$, with weights $W(\mathcal{K}_0)$ initialized by the OWT. Since at every step of the algorithm all remaining contours must have strength greater than or equal to those previously removed, the weight of the contour currently being removed cannot decrease during the merging process. Hence, the constructed region tree has the structure of an indexed hierarchy and can be described by a dendrogram, where the height $H(R)$ of each region R is the value of the dissimilarity at which it first appears. Stated equivalently, $H(R) = W(C)$ where C is the contour whose removal formed R. The hierarchy also yields a metric on $\mathcal{P}_0 \times \mathcal{P}_0$, with the distance between two regions given by the height of the smallest containing segment:

$$D(R_1, R_2) = \min\{H(R) : R_1, R_2 \subseteq R\} \tag{15}$$

This distance satisfies the ultrametric property:

$$D(R_1, R_2) \leq \max(D(R_1, R), D(R, R_2)) \tag{16}$$

since if R is merged with R_1 before R_2, then $D(R_1, R_2) = D(R, R_2)$, or if R is merged with R_2 before R_1, then $D(R_1, R_2) = D(R_1, R)$. As a consequence, the whole

hierarchy can be represented as an Ultrametric Contour Map (UCM) [35], the real-valued image obtained by weighting each boundary by its scale of disappearance.

Figure 14 presents an example of our method. The UCM is a weighted contour image that, by construction, has the remarkable property of producing a set of closed curves for any threshold. Conversely, it is a convenient representation of the region tree since the segmentation at a scale k can be easily retrieved by thresholding the UCM at level k. Since our notion of scale is the average contour strength, the UCM values reflect the contrast between neighboring regions.

4.3 Results

While the OWT-UCM algorithm can use any source of contours for the input $E(x, y, \theta)$ signal (*e.g.* the Canny edge detector before thresholding), we obtain best results by employing the gPb detector [3] introduced in Section 3. We report experiments using both gPb as well as the baseline Canny detector, and refer to the resulting segmentation algorithms as gPb-owt-ucm and Canny-owt-ucm, respectively.

Figures 15 and 16 illustrate results of gPb-owt-ucm on images from the BSDS500. Since the OWT-UCM algorithm produces hierarchical region trees, obtaining a single segmentation as output involves a choice of scale. One possibility is to use a fixed threshold for all images in the dataset, calibrated to provide optimal performance on the training set. We refer to this as the *optimal dataset scale (ODS)*. We also evaluate performance when the optimal threshold is selected by an oracle on a per-image basis. With this choice of *optimal image scale (OIS)*, one naturally obtains even better segmentations.

4.4 Evaluation

To provide a basis of comparison for the OWT-UCM algorithm, we make use of the region merging (Felz-Hutt) [32], Mean Shift [34], Multiscale NCuts [33], and SWA [31], [52] segmentation methods reviewed in Section 2.2. We evaluate each method using the boundary-based precision-recall framework of [2], as well as the Variation of Information, Probabilistic Rand Index, and segment covering criteria discussed in Section 2.3. The BSDS serves as ground-truth for both the boundary and region quality measures, since the human-drawn boundaries are closed and hence are also segmentations.

4.4.1 Boundary Quality

Remember that the evaluation methodology developed by [2] measures detector performance in terms of precision, the fraction of true positives, and recall, the fraction of ground-truth boundary pixels detected. The global F-measure, or harmonic mean of precision and recall at the optimal detector threshold, provides a summary score.

In our experiments, we report three different quantities for an algorithm: the best F-measure on the dataset for a fixed scale (ODS), the aggregate F-measure on the

	BSDS300			BSDS500		
	ODS	OIS	AP	ODS	OIS	AP
Human	0.79	0.79	–	0.80	0.80	–
gPb-owt-ucm	**0.71**	**0.74**	**0.73**	**0.73**	**0.76**	**0.73**
[34] Mean Shift	0.63	0.66	0.54	0.64	0.68	0.56
[33] NCuts	0.62	0.66	0.43	0.64	0.68	0.45
Canny-owt-ucm	0.58	0.63	0.58	0.60	0.64	0.58
[32] Felz-Hutt	0.58	0.62	0.53	0.61	0.64	0.56
[31] SWA	0.56	0.59	0.54	–	–	–
Quad-Tree	0.37	0.39	0.26	0.38	0.39	0.26
gPb	0.70	0.72	0.66	0.71	0.74	0.65
Canny	0.58	0.62	0.58	0.60	0.63	0.58

TABLE 1. **Boundary benchmarks on the BSDS.** Results for seven different segmentation methods (upper table) and two contour detectors (lower table) are given. Shown are the F-measures when choosing an optimal scale for the entire dataset (ODS) or per image (OIS), as well as the average precision (AP). Figures 1, 2, and 17 show the full precision-recall curves for these algorithms. Note that the boundary benchmark has the largest discriminative power among the evaluation criteria, clearly separating the Quad-Tree from all the data-driven methods.

	BSDS300						
	Covering			PRI		VI	
	ODS	OIS	Best	ODS	OIS	ODS	OIS
Human	0.73	0.73	–	0.87	0.87	1.16	1.16
gPb-owt-ucm	**0.59**	**0.65**	**0.75**	**0.81**	**0.85**	**1.65**	**1.47**
[34] Mean Shift	0.54	0.58	0.66	0.78	0.80	1.83	1.63
[32] Felz-Hutt	0.51	0.58	0.68	0.77	0.82	2.15	1.79
Canny-owt-ucm	0.48	0.56	0.66	0.77	0.82	2.11	1.81
[33] NCuts	0.44	0.53	0.66	0.75	0.79	2.18	1.84
[31] SWA	0.47	0.55	0.66	0.75	0.80	2.06	1.75
[29] Total Var.	0.57	–	–	0.78	–	1.81	–
[70] T+B Encode	0.54	–	–	0.78	–	1.86	–
[30] Av. Diss.	0.47	–	–	0.76	–	2.62	–
[30] ChanVese	0.49	–	–	0.75	–	2.54	–
Quad-Tree	0.33	0.39	0.47	0.71	0.75	2.34	2.22

	BSDS500						
	Covering			PRI		VI	
	ODS	OIS	Best	ODS	OIS	ODS	OIS
Human	0.72	0.72	–	0.88	0.88	1.17	1.17
gPb-owt-ucm	**0.59**	**0.65**	**0.74**	**0.83**	**0.86**	**1.69**	**1.48**
[34] Mean Shift	0.54	0.58	0.66	0.79	0.81	1.85	1.64
[32] Felz-Hutt	0.52	0.57	0.69	0.80	0.82	2.21	1.87
Canny-owt-ucm	0.49	0.55	0.66	0.79	0.83	2.19	1.89
[33] NCuts	0.45	0.53	0.67	0.78	0.80	2.23	1.89
Quad-Tree	0.32	0.37	0.46	0.73	0.74	2.46	2.32

TABLE 2. **Region benchmarks on the BSDS.** For each segmentation method, the leftmost three columns report the score of the covering of ground-truth segments according to optimal dataset scale (ODS), optimal image scale (OIS), or Best covering criteria. The rightmost four columns compare the segmentation methods against ground-truth using the Probabilistic Rand Index (PRI) and Variation of Information (VI) benchmarks, respectively. Among the region benchmarks, the covering criterion has the largest dynamic range, followed by PRI and VI.

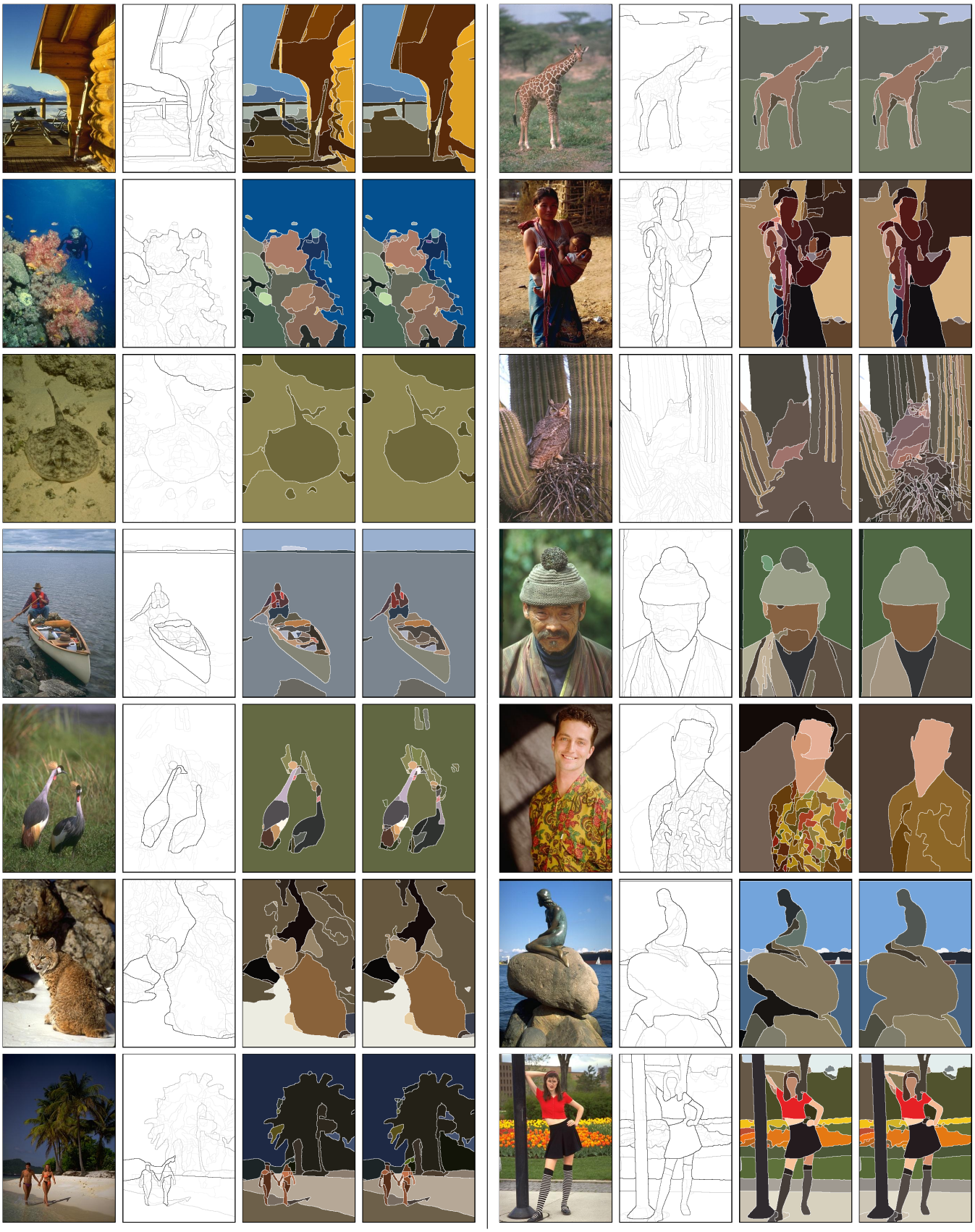

Fig. 15. **Hierarchical segmentation results on the BSDS500. *From Left to Right:*** Image, Ultrametric Contour Map (UCM) produced by gPb-owt-ucm, and segmentations obtained by thresholding at the optimal dataset scale (ODS) and optimal image scale (OIS). All images are from the test set.

Fig. 16. **Additional hierarchical segmentation results on the BSDS500. *From Top to Bottom:*** Image, UCM produced by gPb-owt-ucm, and ODS and OIS segmentations. All images are from the test set.

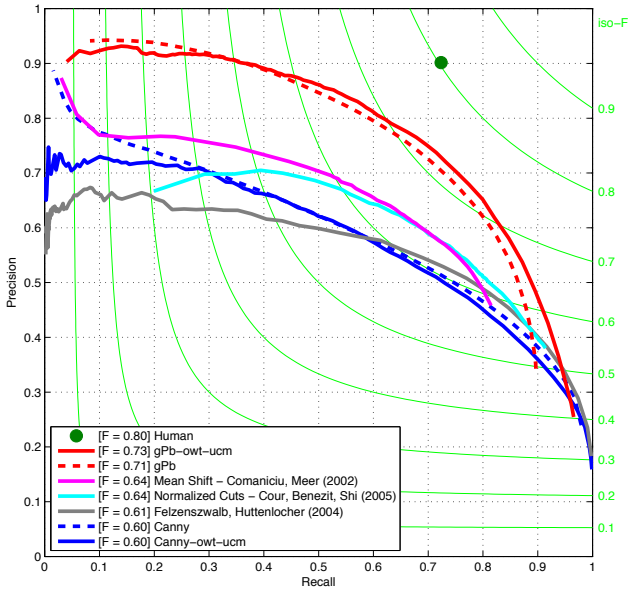

Fig. 17. **Boundary benchmark on the BSDS500.** Comparing boundaries to human ground-truth allows us to evaluate contour detectors [3], [22] (dotted lines) and segmentation algorithms [4], [32], [33], [34] (solid lines) in the same framework. Performance is consistent when going from the BSDS300 (Figures 1 and 2) to the BSDS500 (above). Furthermore, the **OWT-UCM algorithm preserves contour detector quality**. For both gPb and Canny, comparing the resulting segment boundaries to the original contours shows that our OWT-UCM algorithm constructs hierarchical segmentations from contours without losing performance on the boundary benchmark.

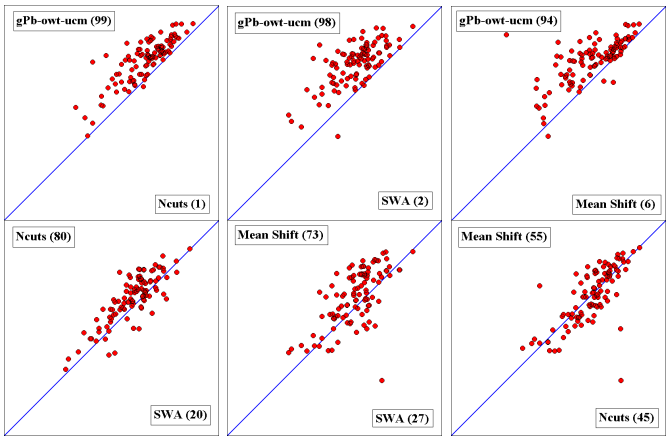

Fig. 18. **Pairwise comparison of segmentation algorithms on the BSDS300.** The coordinates of the red dots are the boundary benchmark scores (F-measures) at the optimal image scale for each of the two methods compared on single images. Boxed totals indicate the number of images where one algorithm is better. For example, the top-left shows gPb-owt-ucm outscores NCuts on $99/100$ images. When comparing with SWA, we further restrict the output of the second method to match the number of regions produced by SWA. All differences are statistically significant except between Mean Shift and NCuts.

	MSRC			PASCAL 2008		
	ODS	OIS	Best	ODS	OIS	Best
gPb-owt-ucm	**0.66**	**0.75**	**0.78**	**0.45**	**0.58**	**0.61**
Canny-owt-ucm	0.57	0.68	0.72	0.40	0.53	0.55

TABLE 3. **Region benchmarks on MSRC and PASCAL 2008.** Shown are scores for the segment covering criteria.

dataset for the best scale in each image (OIS), and the *average precision (AP)* on the full recall range (equivalently, the area under the precision-recall curve). Table 1 shows these quantities for the BSDS. Figures 2 and 17 display the full precision-recall curves on the BSDS300 and BSDS500 datasets, respectively. We find retraining on the BSDS500 to be unnecessary and use the same parameters learned on the BSDS300. Figure 18 presents side by side comparisons of segmentation algorithms.

Of particular note in Figure 17 are pairs of curves corresponding to contour detector output and regions produced by running the OWT-UCM algorithm on that output. The similarity in quality within each pair shows that we can convert contours into hierarchical segmentations without loss of boundary precision or recall.

4.4.2 Region Quality

Table 2 presents region benchmarks on the BSDS. For a family of machine segmentations $\{S_i\}$, associated with different scales of a hierarchical algorithm or different sets of parameters, we report three scores for the covering of the ground-truth by segments in $\{S_i\}$. These correspond to selecting covering regions from the segmentation at a universal fixed scale (ODS), a fixed scale per image (OIS), or from any level of the hierarchy or collection $\{S_i\}$ (Best). We also report the Probabilistic Rand Index and Variation of Information benchmarks.

While the relative ranking of segmentation algorithms remains fairly consistent across different benchmark criteria, the boundary benchmark (Table 1 and Figure 17) appears most capable of discriminating performance. This observation is confirmed by evaluating a fixed hierarchy of regions such as the Quad-Tree (with 8 levels). While the boundary benchmark and segmentation covering criterion clearly separate it from all other segmentation methods, the gap narrows for the Probablilistic Rand Index and the Variation of Information.

4.4.3 Additional Datasets

We concentrated experiments on the BSDS because it is the most complete dataset available for our purposes, has been used in several publications, and has the advantage of providing multiple human-labeled segmentations per image. Table 3 reports the comparison between Canny-owt-ucm and gPb-owt-ucm on two other publicly available datasets:

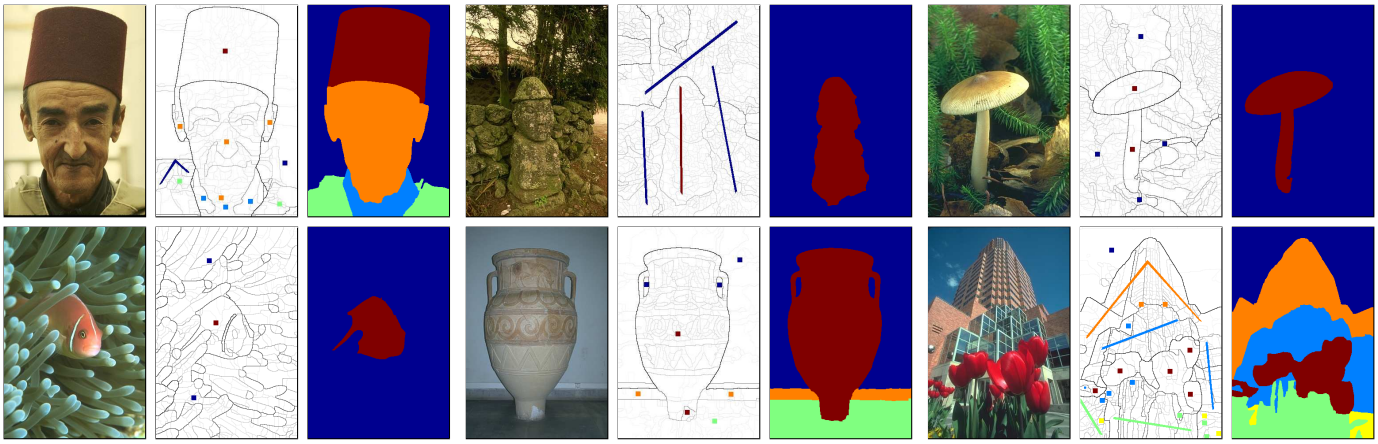

Fig. 19. **Interactive segmentation.** ***Left:*** Image. ***Middle:*** UCM produced by gPb-owt-ucm (grayscale) with additional user annotations (color dots and lines). ***Right:*** The region hierarchy defined by the UCM allows us to automatically propagate annotations to unlabeled segments, resulting in the desired labeling of the image with minimal user effort.

- **MSRC** [71]
  The MSRC object recognition database is composed of 591 natural images with objects belonging to 21 classes. We evaluate performance using the ground-truth object instance labeling of [11], which is cleaner and more precise than the original data.

- **PASCAL 2008** [8]
  We use the train and validation sets of the segmentation task on the 2008 PASCAL segmentation challenge, composed of 1023 images. This is one of the most difficult and varied datasets for recognition. We evaluate performance with respect to the object instance labels provided. Note that only objects belonging to the 20 categories of the challenge are labeled, and 76% of all pixels are unlabeled.

4.4.4 Summary

The gPb-owt-ucm segmentation algorithm offers the best performance on every dataset and for every benchmark criterion we tested. In addition, it is straight-forward, fast, has no parameters to tune, and, as discussed in the following sections, can be adapted for use with top-down knowledge sources.

5 Interactive Segmentation

Until now, we have only discussed fully automatic image segmentation. Human assisted segmentation is relevant for many applications, and recent approaches rely on the graph-cuts formalism [72], [73], [74] or other energy minimization procedure [75] to extract foreground regions.

For example, [72] cast the task of determining binary foreground/background pixel assignments in terms of a cost function with both unary and pairwise potentials. The unary potentials encode agreement with estimated foreground or background region models and the pairwise potentials bias neighboring pixels not separated by a strong boundary to have the same label. They transform this system into an equivalent minimum cut/maximum flow graph partitioning problem through the addition of a source node representing the foreground and a sink node representing the background. Edge weights between pixel nodes are defined by the pairwise potentials, while the weights between pixel nodes and the source and sink nodes are determined by the unary potentials. User-specified hard labeling constraints are enforced by connecting a pixel to the source or sink with sufficiently large weight. The minimum cut of the resulting graph can be computed efficiently and produces a cost-optimizing assignment.

It turns out that the segmentation trees generated by the OWT-UCM algorithm provide a natural starting point for user-assisted refinement. Following the procedure of [76], we can extend a partial labeling of regions to a full one by assigning to each unlabeled region the label of its closest labeled region, as determined by the ultrametric distance (15). Computing the full labeling is simply a matter of propagating information in a single pass along the segmentation tree. Each unlabeled region receives the label of the first labeled region merged with it. This procedure, illustrated in Figure 19, allows a user to obtain high quality results with minimal annotation.

6 Multiscale for Object Analysis

Our contour detection and segmentation algorithms capture multiscale information by combining local gradient cues computed at three different scales, as described in Section 3.2. We did not see any performance benefit on the BSDS by using additional scales. However, this fact is not an invitation to conclude that a simple combination of a limited range of local cues is a sufficient solution to the problem of multiscale image analysis. Rather, it is a statement about the nature of the BSDS. The fixed resolution of the BSDS images and the inherent photographic bias of the dataset lead to the situation in

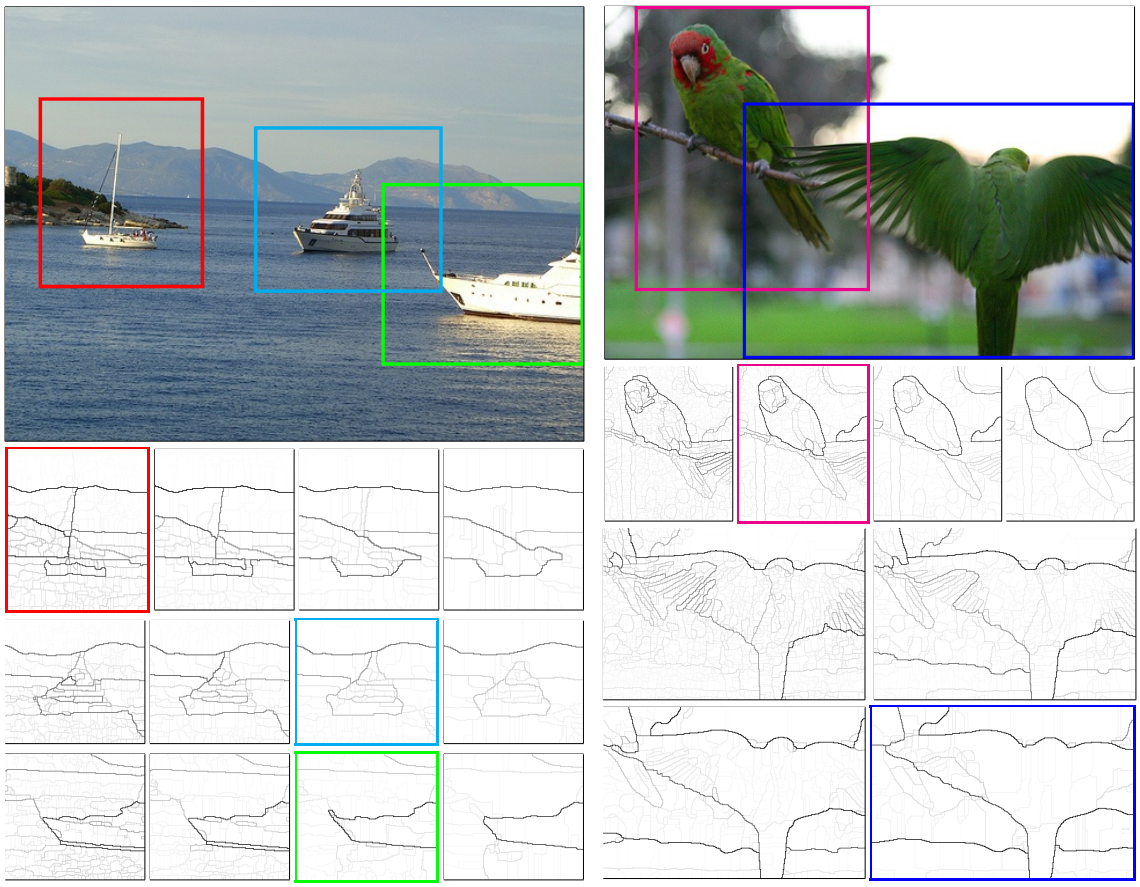

Fig. 20. **Multiscale segmentation for object detection.** ***Top:*** Images from the PASCAL 2008 dataset, with objects outlined at ground-truth locations. ***Detailed Views:*** For each window, we show the boundaries obtained by running the entire gPb-owt-ucm segmentation algorithm at multiple scales. Scale increases by factors of 2 moving from left to right (and top to bottom for the blue window). The total scale range is thus larger than the three scales used internally for each segmentation. ***Highlighted Views:*** The highlighted scale best captures the target object's boundaries. Note the link between this scale and the absolute size of the object in the image. For example, the small sailboat (red outline) is correctly segmented only at the finest scale. In other cases (*e.g.* parrot, magenta outline), bounding contours appear across several scales, but informative internal contours are scale sensitive. A window-based object detector could learn and exploit an optimal coupling between object size and segmentation scale.

which a small range of scales captures the boundaries humans find important.

Dealing with the full variety one expects in high resolution images of complex scenes requires more than a naive weighted average of signals across the scale range. Such an average would blur information, resulting in good performance for medium-scale contours, but poor detection of both fine-scale and large-scale contours. Adaptively selecting the appropriate scale at each location in the image is desirable, but it is unclear how to estimate this robustly using only bottom-up cues.

For some applications, in particular object detection, we can instead use a top-down process to guide scale selection. Suppose we wish to apply a classifier to determine whether a subwindow of the image contains an instance of a given object category. We need only report a positive answer when the object completely fills the subwindow, as the detector will be run on a set of windows densely sampled from the image. Thus, we know the size of the object we are looking for in each window and hence the scale at which contours belonging to the object would appear. Varying the contour scale with the window size produces the best input signal for the object detector. Note that this procedure does not prevent the object detector itself from using multiscale information, but rather provides the correct central scale.

As each segmentation internally uses gradients at three scales, $[\frac{\sigma}{2}, \sigma, 2\sigma]$, by stepping by a factor of 2 in scale between segmentations, we can reuse shared local cues. The globalization stage (sPb signal) can optionally be customized for each window by computing it using only a limited surrounding image region. This strategy, used here, results in more work overall (a larger number of simpler globalization problems), which can be mitigated by not sampling sPb as densely as one samples windows.

Figure 20 shows an example using images from the PASCAL dataset. Bounding boxes displayed are slightly larger than each object to give some context. Multiscale segmentation shows promise for detecting fine-scale objects in scenes as well as making salient details available together with large-scale boundaries.

APPENDIX
EFFICIENT COMPUTATION

Computing the oriented gradient of histograms (Figure 4) directly as outlined in Section 3.1 is expensive. In particular, for an N pixel image and a disc of radius r it takes $O(Nr^2)$ time to compute since a region of area $O(r^2)$ is examined at every pixel location. This entire procedure is repeated 32 times (4 channels with 8 orientations) for each of 3 choices of r (the cost of the largest scale dominates the time complexity). Martin *et al.* [2] suggest ways to speed up this computation, including incremental updating of the histograms as the disc is swept across the image. However, this strategy still requires $O(Nr)$ time. We present an algorithm for the oriented gradient of histograms computation that runs in $O(N)$ time, independent of the radius r.

Following Figure 21, we can approximate each half-disc by a series of rectangles. It turns out that a single rectangle is a sufficiently good approximation for our purposes (in principle, we can always choose a fixed number of rectangles to achieve any specified accuracy). Now, instead of rotating our rectangular regions, we pre-rotate the image so that we are concerned with computing a histogram of the values within axis-aligned rectangles. This can be done in time independent of the size of the rectangle using integral images.

We process each histogram bin separately. Let I denote the rotated intensity image and let $I^b(x, y)$ be 1 if $I(x, y)$ falls in histogram bin b and 0 otherwise. Compute the integral image J^b as the cumulative sum across rows of the cumulative sum across columns of I^b. The total energy in an axis-aligned rectangle with points P, Q, R, and S as its upper-left, upper-right, lower-left, and lower-right corners, respectively, falling in histogram bin b is:

$$h(b) = J^b(P) + J^b(S) - J^b(Q) - J^b(R) \qquad (17)$$

It takes $O(N)$ time to pre-rotate the image, and $O(N)$ to compute each of the $O(B)$ integral images, where B is the number of bins in the histogram. Once these are computed, there is $O(B)$ work per rectangle, of which there are $O(N)$. Rotating the output back to the original coordinate frame takes an additional $O(N)$ work. The total complexity is thus $O(NB)$ instead of $O(Nr^2)$ (actually instead of $O(Nr^2 + NB)$ since we always had to compute χ^2 distances between N histograms). Since B is a fixed constant, the computation time no longer grows as we increase the scale r.

This algorithm runs in $O(NB)$ time as long as we use at most a constant number of rectangular boxes to

Fig. 21. **Efficient computation of the oriented gradient of histograms.** ***Left:*** The two half-discs of interest can be approximated arbitrarily well by a series of rectangular boxes. We found a single box of equal area to the half-disc to be a sufficient approximation. ***Middle:*** Replacing the circular disc of Figure 4 with the approximation reduces the problem to computing the histograms within rectangular regions. ***Right:*** Instead of rotating the rectangles, rotate the image and use the integral image trick to compute the histograms efficiently. Rotate the final result to map it back to the original coordinate frame.

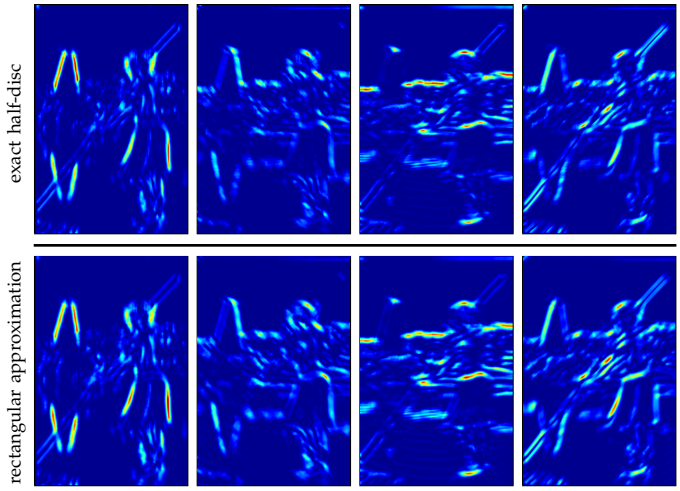

Fig. 22. **Comparison of half-disc and rectangular regions for computing the oriented gradient of histograms.** ***Top Row:*** Results of using the $O(Nr^2)$ time algorithm to compute the difference of histograms in oriented half-discs at each pixel. Shown is the output for processing the brightness channel displayed in Figure 21 using a disc of radius $r = 10$ pixels at four distinct orientations (one per column). N is the total number of pixels. ***Bottom Row:*** Approximating each half-disc with a single rectangle (of height 9 pixels so that the rectangle area best matches the disc area), as shown in Figure 21, and using integral histograms allows us to compute nearly identical results in only $O(N)$ time. In both cases, we show the raw histogram difference output before application of a smoothing filter in order to clearly demonstrate the similarity of the results.

approximate each half-disc. For an intuition as to why a single rectangle turns out to be sufficient, look again at the overlap of the rectangle with the half disc in the lower left of Figure 21. The majority of the pixels used in forming the histogram lie within both the rectangle and the disc, and those pixels that differ are far from the center of the disc (the pixel at which we are computing the gradient). Thus, we are only slightly changing the shape of the region we use for context around each pixel. Figure 22 shows that the result using the single rectangle approximation is visually indistinguishable from that using the original half-disc.

Note that the same image rotation technique can be used for computing convolutions with any oriented separable filter, such as the oriented Gaussian derivative filters used for textons (Figure 5) or the second-order Savitzky-Golay filters used for spatial smoothing of our oriented gradient output. Rotating the image, convolving with two 1D filters, and inverting the rotation is more efficient than convolving with a rotated 2D filter. Moreover, in this case, no approximation is required as these operations are equivalent up to the numerical accuracy of the interpolation done when rotating the image. This means that all of the filtering performed as part of the local cue computation can be done in $O(Nr)$ time instead of $O(Nr^2)$ time where here $r = max(w, h)$ and w and h are the width and height of the 2D filter matrix. For large r, the computation time can be further reduced by using the Fast Fourier Transform to calculate the convolution.

The entire local cue computation is also easily parallelized. The image can be partitioned into overlapping subimages to be processed in parallel. In addition, the 96 intermediate results (3 scales of 4 channels with 8 orientations) can all be computed in parallel as they are independent subproblems. Catanzaro *et al.* [77] have created a parallel GPU implementation of our gPb contour detector. They also exploit the integral histogram trick introduced here, with the single rectangle approximation, while replicating our precision-recall performance curve on the BSDS benchmark. The speed improvements due to both the reduction in computational complexity and parallelization make our gPb contour detector and gPb-owt-ucm segmentation algorithm practical tools.

REFERENCES

[1] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," *ICCV*, 2001.

[2] D. Martin, C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using local brightness, color and texture cues," *PAMI*, 2004.

[3] M. Maire, P. Arbeláez, C. Fowlkes, and J. Malik, "Using contours to detect and localize junctions in natural images," *CVPR*, 2008.

[4] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "From contours to regions: An empirical evaluation," *CVPR*, 2009.

[5] R. Unnikrishnan, C. Pantofaru, and M. Hebert, "Toward objective evaluation of image segmentation algorithms," *PAMI*, 2007.

[6] M. Meila, "Comparing clusterings: An axiomatic view," *ICML*, 2005.

[7] A. Y. Yang, J. Wright, Y. Ma, and S. S. Sastry, "Unsupervised segmentation of natural images via lossy data compression," *CVIU*, 2008.

[8] M. Everingham, L. van Gool, C. Williams, J. Winn, and A. Zisserman, "PASCAL 2008 Results," http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html, 2008.

[9] D. Hoiem, A. A. Efros, and M. Hebert, "Geometric context from a single image," *ICCV*, 2005.

[10] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie, "Objects in context," *ICCV*, 2007.

[11] T. Malisiewicz and A. A. Efros, "Improving spatial support for objects via multiple segmentations," *BMVC*, 2007.

[12] N. Ahuja and S. Todorovic, "Connected segmentation tree: A joint representation of region layout and hierarchy," *CVPR*, 2008.

[13] A. Saxena, S. H. Chung, and A. Y. Ng, "3-d depth reconstruction from a single still image," *IJCV*, 2008.

[14] T. Brox, C. Bregler, and J. Malik, "Large displacement optical flow," *CVPR*, 2009.

[15] C. Gu, J. Lim, P. Arbeláez, and J. Malik, "Recognition using regions," *CVPR*, 2009.

[16] J. Lim, P. Arbeláez, C. Gu, and J. Malik, "Context by region ancestry," *ICCV*, 2009.

[17] L. G. Roberts, "Machine perception of three-dimensional solids," *In Optical and Electro-Optical Information Processing*, J. T. Tippett et al. Eds. Cambridge, MA: MIT Press, 1965.

[18] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[19] J. M. S. Prewitt, "Object enhancement and extraction," *In Picture Processing and Psychopictorics*, B. Lipkin and A. Rosenfeld. Eds. Academic Press, New York, 1970.

[20] D. C. Marr and E. Hildreth, "Theory of edge detection," *Proceedings of the Royal Society of London*, 1980.

[21] P. Perona and J. Malik, "Detecting and localizing edges composed of steps, peaks and roofs," *ICCV*, 1990.

[22] J. Canny, "A computational approach to edge detection," *PAMI*, 1986.

[23] X. Ren, C. Fowlkes, and J. Malik, "Scale-invariant contour completion using conditional random fields," *ICCV*, 2005.

[24] Q. Zhu, G. Song, and J. Shi, "Untangling cycles for contour grouping," *ICCV*, 2007.

[25] P. Felzenszwalb and D. McAllester, "A min-cover approach for finding salient curves," *POCV*, 2006.

[26] J. Mairal, M. Leordeanu, F. Bach, M. Hebert, and J. Ponce, "Discriminative sparse image models for class-specific edge detection and image interpretation," *ECCV*, 2008.

[27] P. Dollar, Z. Tu, and S. Belongie, "Supervised learning of edges and object boundaries," *CVPR*, 2006.

[28] X. Ren, "Multi-scale improves boundary detection in natural images," *ECCV*, 2008.

[29] M. Donoser, M. Urschler, M. Hirzer, and H. Bischof, "Saliency driven total variational segmentation," *ICCV*, 2009.

[30] L. Bertelli, B. Sumengen, B. Manjunath, and F. Gibou, "A variational framework for multi-region pairwise similarity-based image segmentation," *PAMI*, 2008.

[31] E. Sharon, M. Galun, D. Sharon, R. Basri, and A. Brandt, "Hierarchy and adaptivity in segmenting visual scenes," *Nature*, vol. 442, pp. 810–813, 2006.

[32] P. Felzenszwalb and D. Huttenlocher, "Efficient graph-based image segmentation," *IJCV*, 2004.

[33] T. Cour, F. Benezit, and J. Shi, "Spectral segmentation with multiscale graph decomposition." *CVPR*, 2005.

[34] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *PAMI*, 2002.

[35] P. Arbeláez, "Boundary extraction in natural images using ultrametric contour maps," *POCV*, 2006.

[36] M. C. Morrone and R. Owens, "Feature detection from local energy," *Pattern Recognition Letters*, 1987.

[37] W. T. Freeman and E. H. Adelson, "The design and use of steerable filters," *PAMI*, 1991.

[38] T. Lindeberg, "Edge detection and ridge detection with automatic scale selection," *IJCV*, 1998.

[39] Z. Tu, "Probabilistic boosting-tree: Learning discriminative models for classification, recognition, and clustering," *ICCV*, 2005.

[40] P. Parent and S. W. Zucker, "Trace inference, curvature consistency, and curve detection," *PAMI*, 1989.

[41] L. R. Williams and D. W. Jacobs, "Stochastic completion fields: A neural model of illusory contour shape and salience," *ICCV*, 1995.

[42] J. Elder and S. Zucker, "Computing contour closures," *ECCV*, 1996.
[43] Y. Weiss, "Correctness of local probability propagation in graphical models with loops," *Neural Computation*, 2000.
[44] S. Belongie, C. Carson, H. Greenspan, and J. Malik, "Color and texture-based image segmentation using EM and its application to content-based image retrieval," *ICCV*, pp. 675–682, 1998.
[45] J. Shi and J. Malik, "Normalized cuts and image segmentation," *PAMI*, 2000.
[46] J. Malik, S. Belongie, T. Leung, and J. Shi, "Contour and texture analysis for image segmentation," *IJCV*, 2001.
[47] D. Tolliver and G. L. Miller, "Graph partitioning by spectral rounding: Applications in image segmentation and clustering," *CVPR*, 2006.
[48] F. R. K. Chung, *Spectral Graph Theory*. American Mathematical Society, 1997.
[49] C. Fowlkes and J. Malik, "How much does globalization help segmentation?" UC Berkeley, Tech. Rep. CSD-04-1340, 2004.
[50] S. Wang, T. Kubota, J. M. Siskind, and J. Wang, "Salient closed boundary extraction with ratio contour," *PAMI*, 2005.
[51] S. X. Yu, "Segmentation induced by scale invariance," *CVPR*, 2005.
[52] S. Alpert, M. Galun, R. Basri, and A. Brandt, "Image segmentation by probabilistic bottom-up aggregation and cue integration," *CVPR*, 2007.
[53] D. Mumford and J. Shah, "Optimal approximations by piecewise smooth functions, and associated variational problems," *Communications on Pure and Applied Mathematics*, pp. 577–684, 1989.
[54] J. M. Morel and S. Solimini, *Variational Methods in Image Segmentation*. Birkhäuser, 1995.
[55] G. Koepfler, C. Lopez, and J. Morel, "A multiscale algorithm for image segmentation by variational method," *SIAM Journal on Numerical Analysis*, 1994.
[56] T. Chan and L. Vese, "Active contours without edges," *IP*, 2001.
[57] L. Vese and T. Chan, "A multiphase level set framework for image segmentation using the mumford and shah model," *IJCV*, 2002.
[58] S. Osher and J. Sethian, "Fronts propagation with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations," *Journal of Computational Physics*, 1988.
[59] J. A. Sethian, *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999.
[60] T. Pock, D. Cremers, H. Bischof, and A. Chambolle, "An algorithm for minimizing the piecewise smooth mumford-shah functional," *ICCV*, 2009.
[61] F. J. Estrada and A. D. Jepson, "Benchmarking image segmentation algorithms," *IJCV*, 2009.
[62] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the American Statistical Association*, vol. 66, pp. 846–850, 1971.
[63] A. Savitzky and M. J. E. Golay, "Smoothing and differentiation of data by simplified least squares procedures," *Analytical Chemistry*, 1964.
[64] C. Fowlkes, D. Martin, and J. Malik, "Learning affinity functions for image segmentation: Combining patch-based and gradient-based approaches," *CVPR*, 2003.
[65] T. Leung and J. Malik, "Contour continuity in region-based image segmentation," *ECCV*, 1998.
[66] S. Belongie and J. Malik, "Finding boundaries in natural images: A new method using point descriptors and area completion," *ECCV*, 1998.
[67] X. Ren and J. Malik, "Learning a classification model for segmentation," *ICCV*, 2003.
[68] S. Beucher and F. Meyer, *Mathematical Morphology in Image Processing*. Marcel Dekker, 1992, ch. 12.
[69] L. Najman and M. Schmitt, "Geodesic saliency of watershed contours and hierarchical segmentation," *PAMI*, 1996.
[70] S. Rao, H. Mobahi, A. Yang, S. Sastry, and Y. Ma, "Natural image segmentation with adaptive texture and boundary encoding," *ACCV*, 2009.
[71] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation," *ECCV*, 2006.
[72] Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images." *ICCV*, 2001.
[73] C. Rother, V. Kolmogorov, and A. Blake, ""Grabcut": Interactive foreground extraction using iterated graph cuts," *SIGGRAPH*, 2004.
[74] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum, "Lazy snapping," *SIGGRAPH*, 2004.
[75] S. Bagon, O. Boiman, and M. Irani, "What is a good image segment? A unified approach to segment extraction," *ECCV*, 2008.
[76] P. Arbeláez and L. Cohen, "Constrained image segmentation from hierarchical boundaries," *CVPR*, 2008.
[77] B. Catanzaro, B.-Y. Su, N. Sundaram, Y. Lee, M. Murphy, and K. Keutzer, "Efficient, high-quality image contour detection," *ICCV*, 2009.

Pablo Arbeláez received a PhD with honors in Applied Mathematics from the Université Paris Dauphine in 2005. He is a Research Scholar with the Computer Vision group at UC Berkeley since 2007. His research interests are in computer vision, where he has worked on a number of problems, including perceptual grouping, object recognition and the analysis of biological images.

Michael Maire received a BS with honors from the California Institute of Technology in 2003 and a PhD in Computer Science from the University of California, Berkeley in 2009. He is currently a Postdoctoral Scholar in the Department of Electrical Engineering at Caltech. His research interests are in computer vision as well as its use in biological image and video analysis.

Charless Fowlkes received a BS with honors from Caltech in 2000 and a PhD in Computer Science from the University of California, Berkeley in 2005, where his research was supported by a US National Science Foundation Graduate Research Fellowship. He is currently an Assistant Professor in the Department of Computer Science at the University of California, Irvine. His research interests are in computer and human vision, and in applications to biological image analysis.

Jitendra Malik was born in Mathura, India in 1960. He received the B.Tech degree in Electrical Engineering from Indian Institute of Technology, Kanpur in 1980 and the PhD degree in Computer Science from Stanford University in 1985. In January 1986, he joined the university of California at Berkeley, where he is currently the Arthur J. Chick Professor in the Computer Science Division, Department of Electrical Engineering and Computer Sciences. He is also on the faculty of the Cognitive Science and Vision Science groups. During 2002-2004 he served as the Chair of the Computer Science Division and during 2004-2006 as the Department Chair of EECS. He serves on the advisory board of Microsoft Research India, and on the Governing Body of IIIT Bangalore.

His current research interests are in computer vision, computational modeling of human vision and analysis of biological images. His work has spanned a range of topics in vision including image segmentation, perceptual grouping, texture, stereopsis and object recognition with applications to image based modeling and rendering in computer graphics, intelligent vehicle highway systems, and biological image analysis. He has authored or co-authored more than a hundred and fifty research papers on these topics, and graduated twenty-five PhD students who occupy prominent places in academia and industry. According to Google Scholar, four of his papers have received more than a thousand citations.

He received the gold medal for the best graduating student in Electrical Engineering from IIT Kanpur in 1980 and a Presidential Young Investigator Award in 1989. At UC Berkeley, he was selected for the Diane S. McEntyre Award for Excellence in Teaching in 2000, a Miller Research Professorship in 2001, and appointed to be the Arthur J. Chick Professor in 2002. He received the Distinguished Alumnus Award from IIT Kanpur in 2008. He was awarded the Longuet-Higgins Prize for a contribution that has stood the test of time twice, in 2007 and in 2008. He is a fellow of the IEEE and the ACM.