Natural Image Segmentation with Adaptive Texture and Boundary Encoding

Shankar R. Rao[†], Hossein Mobahi[‡], Allen Y. Yang[◇], S. Shankar Sastry[◇], Yi Ma[†*]

[†]Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign

[‡]Department of Computer Science
University of Illinois at Urbana-Champaign

[◇]Department of Electrical Engineering and Computer Science
University of California at Berkeley

[*]Visual Computing Group
Microsoft Research Asia

April 17, 2009

Abstract

We present a novel algorithm for unsupervised segmentation of natural images that harnesses the principle of minimum description length (MDL). Our method is based on observations that a homogeneously textured region of a natural image can be well modeled by a Gaussian distribution and the region boundary can be effectively coded by an adaptive chain code. The optimal segmentation of an image is the one that gives the shortest coding length for encoding all textures and boundaries in the image, and is obtained via an agglomerative clustering process applied to a hierarchy of decreasing window sizes. The optimal segmentation also provides an accurate estimate of the overall coding length and hence the true entropy of the image. We test our algorithm on two publicly available databases: Berkeley Segmentation Dataset and MSRC Object Recognition Database. It achieves state-of-the-art segmentation results compared to other popular methods.

1 Introduction

The task of partitioning a natural image into regions with homogeneous texture, commonly referred to as *image segmentation*, is widely accepted as a crucial first step for high-level image understanding, significantly reducing the complexity of content analysis of images. Image segmentation and its higher-level applications are largely designed to emulate functionalities of human visual perception (e.g., object recognition and scene understanding), and hence dominant criteria for measuring segmentation performance are based on qualitative and quantitative comparisons with human segmentation results. In the literature, investigators have explored several important models and principles that can lead to good image segmentation:

1. Different texture regions of a natural image admit a mixture model. For example, Normalized Cuts (NC) [1] and F&H [2] formulate the segmentation as a graph-cut problem, while Mean Shift (MS) [3] seeks a partition of an color image based on different modes within the estimated empirical distribution.

2. Region contours/edges convey important information about the saliency of the objects in the image and their shapes [4, 5, 6, 7]. Several recent methods have been proposed to combine the cues of homogeneous color and texture with the cue of contours in the segmentation process [8, 9, 10].

3. The properties of local features (including texture and edges) usually do not share the same level of homogeneity at the same spatial scale. Thus, salient image regions can only be extracted from a hierarchy of image features under multiple resolutions [11, 12, 13].

Despite much work in this area, good image segmentation remains elusive to obtain for practitioners, for the following two reasons: 1. There is little consensus on what criteria should be used to evaluate the quality of image segmentations. It is difficult to strike a good balance between objective measures that depend solely on the intrinsic statistics of imagery data and subjective measures that try to empirically model human perception. 2. In the search for objective measures, there has been a lack of consensus on good models for a unified representation of image segments including both their textures and contours.

Recently an *objective* metric based on the notion of lossy *minimum description length* (MDL) has been proposed for evaluating clustering of general mixed data [14]. The basic idea is that, given a potentially mixed data set, the "optimal segmentation" is the one that, over all possible segmentations, minimizes the coding length of the data, subject to a given quantization error. For data drawn from a mixture of Gaussians, the optimal segmentation can often be found efficiently using an agglomerative clustering approach. The MDL principle and the new clustering method have later been applied to the segmentation of natural images, known as *compression-based texture merging* (CTM) [13]. According to several popular segmentation indices, e.g., probabilistic Rand index (PRI) and variation of information (VOI), this approach has proven to be highly effective for imitating human segmentation of natural images. Preliminary success of this approach leads to the following important question: *To what extent is segmentation obtained by image compression consistent with human perception?*

However, although the CTM method utilizes the idea of data compression, it does not exactly seek to compress the image *per se*. First, it "compresses" feature vectors or windows extracted around all pixels by grouping them into clusters as a mixture of Gaussian models. As a result, the final coding length is highly *redundant* due to severe overlap between windows of adjacent pixels, and has no direct relation to the true entropy of the image. Second, the segmentation result encodes the membership of pixels using a Huffman code that does not taking into account of spatial adjacency of pixels nor smoothness of boundaries. Thus, CTM does not give a good estimate of the true entropy of the image and its success cannot be used to justify a strong connection between image segmentation and image compression.

Contributions. In this paper, we contend that, much better segmentation results can be obtained if we follow more closely the principle of image compression, by correctly counting only the necessary bits needed to encode a natural image for both the texture and boundaries. The proposed algorithm precisely estimates the coding length needed to encode the texture of each region based on the rate distortion of its probabilistic distribution and the number of *non-overlapping* windows inside. In order to adapt to the different scales and shapes of texture regions in the image, a hierarchy of multiple window sizes is incorporated in the segmentation process. The algorithm further encodes the boundary information of each homogeneous texture region by carefully counting the number of bits needed to encode the boundary with an adaptive chain code.

Based on the MDL principle, the optimal segmentation of an image is defined as the one that minimizes its total coding length, in this case a close approximation to the true entropy of the image. With any fixed quantization, the final coding length gives a purely objective measure for how good the segmentation is in terms of the level of image compression. We conduct extensive experiments to compare the results with human segmentation, using the Berkeley Segmentation Dataset (BSD) [15] and MSRC Object Recognition Database (MSRC) [16]. Although our method is conceptually simple and the measure used is purely objective, the segmentation results match extremely well with those by human, exceeding or competing with the best segmentation algorithms.

2 Adaptive Texture and Boundary Encoding

In this section, we present a unified information-theoretic framework to encode both the texture and boundary information of a natural image. The implementation of the algorithm for adaptive image segmentation and the experiments to validate its performance will be presented in Sections 3 and 4.

2.1 Gaussianity of Image Textures

First, we discuss how to construct texture vectors that represent homogeneous textures in image segments. Given an image in RGB format, we convert it to the $L^*a^*b^*$ color space. It has been noted in the literature that such a color metric better approximates the perceptually uniform color space. In order to capture the variation of a local *texton*, one can directly apply a $w \times w$ cut-off window around a pixel across the three color channels, and stack the color values inside the window in a vector form as in [13]. [1]

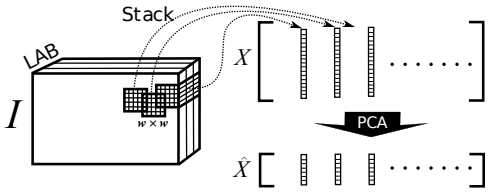


Figure 1: We construct features by stacking the $w \times w$ windows around all pixels of a $L^*a^*b^*$ image I into a data matrix X and then using PCA.

Figure 1 (left) illustrates our process for constructing features. Let the w-neighborhood $\mathcal{W}_w(p)$ be the set of all pixels in a $w \times w$ window centered at pixel p. We construct a set of features X by taking the w-neighborhood around each pixel in I, and then stacking each window as a column vector:

$$X \doteq \{\boldsymbol{x}_p \in \mathbb{R}^{3w^2} : \boldsymbol{x}_p = \mathcal{W}_w(p)^S \text{ for } p \in I\}. \tag{1}$$

For ease of computation, we reduce the dimensionality of these features by projecting the set of all features X onto their first D principal components. We denote the set of features with reduced dimensionality as $\hat{X}$. We have observed that for many natural images, the first eight principal components of X contain over 99% of the energy. In this paper, we choose to assign $D = 8$.

Over the years, there have been many proposed methods to model the representation of image textures in natural images. One model that has been shown to be successful in encoding textures both empirically and theoretically is the Gaussian *Mesh Markov Model* (MMM) [18]. Particularly in texture synthesis, the Gaussian MMM provides consistent estimates of the joint distribution of the pixels in a window, which then can be used to fill in missing texture patches via a simple nonparametric scheme [19].

However, to determine the optimal compression rate for samples from a distribution, one must know the rate-distortion function of that distribution [13]. Unfortunately, the rate-distortion function for MMMs is, to our knowledge, not known in closed form, and difficult to estimate empirically. Over all distributions with the same variance, it is known that the Gaussian distribution will have the highest rate-distortion, and is in this sense, the worst case distribution for compression. Thus by using the rate-distortion for a Gaussian distribution, we obtain an upper bound for the true coding length of the MMM.

In the following, we provide an empirical experiment to determine in which color space (RGB or $L^*a^*b^*$) feature windows from a region with homogeneous texture are better fit by a Gaussian distribution. We use as the ground truth training images from the BSD that were manually segmented by humans. Given the feature vectors within each region, we model the distribution both parametrically and non-parametrically. The parametric model Q is a multivariate normal distribution whose parameters are estimated from the samples using maximum likelihood. The non-parametric model P is obtained by kernel density estimation. If the true distribution is indeed normal, then P and Q should be very similar. Thus the KL divergence $D_{KL}(P \parallel Q)$ can be used to measure the non-Gaussianity of the distribution. The overall non-Gaussianity of each image is simply the average of the non-Gaussianity over all regions. We repeat the above procedure for the entire manually segmented image dataset and estimate the distribution of KL divergence by kernel density estimation for RGB and $L^*a^*b^*$ spaces. As Figure 2 shows, between the two color metrics, $L^*a^*b^*$ has lower mean and standard deviation and hence is better modeled by a Gaussian distribution.

[1] Another popular approach for constructing texture vectors is to use multivariate responses of a fixed 2-D texture filter bank. A previous study [17] has argued that the difference in segmentation results between the two approaches is small, and yet it is more expensive to compute 2-D filter bank responses.

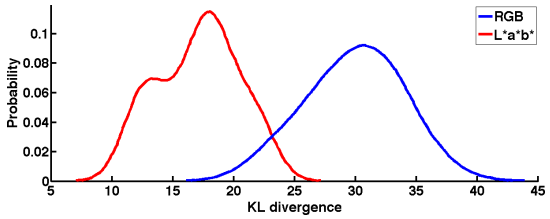


Figure 2: (color) KL divergence of RGB and $L^*a^*b^*$ windows from a true Gaussian distribution.

2.2 Adaptive Texture Encoding

We now describe encoding the texture vectors based on the lossy MDL principle. First, we consider a single region R with N pixels. Based on [13], for a fixed quantization error ε, the expected number of bits needed to code the set of N feature windows $\hat{X}$ up to distortion ε^2 is given by:

$$L_\varepsilon(\hat{X}) \doteq \underbrace{\tfrac{D}{2}\log_2\det(I+\tfrac{D}{\varepsilon^2}\Sigma)}_{\text{codebook}} + \underbrace{\tfrac{N}{2}\log_2\det(I+\tfrac{D}{\varepsilon^2}\Sigma)}_{\text{data}} + \underbrace{\tfrac{D}{2}\log_2(1+\tfrac{\|\boldsymbol{\mu}\|^2}{\varepsilon^2})}_{\text{mean}}, \tag{2}$$

where $\boldsymbol{\mu}$ and Σ are the mean and covariance of the feature windows in $\hat{X}$. Equation (2) is the sum of three coding-lengths: the D Gaussian principal vectors as the codebook, the N windows w.r.t. that codebook, and the mean of the Gaussian distribution.

The coding length function (2) is uniquely determined by the mean and covariance $(\boldsymbol{\mu}, \Sigma)$. To estimate them empirically, we need to exclude the windows that cross the boundary of R (as shown in Figure 3). Such windows contain textures from the adjacent regions, which cannot be well modeled by a single Gaussian as the interior windows. Hence, the empirical mean $\hat{\boldsymbol{\mu}}_w$ and covariance $\hat{\Sigma}_w$ of R from are only estimated from the *interior* of R:

$$\mathcal{I}_w(R) \doteq \{p \in R : q \in R, \forall q \in \mathcal{W}_w(p)\}. \tag{3}$$

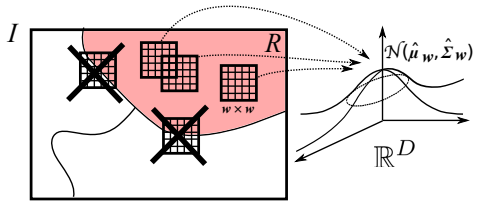


Figure 3: Only windows from the interior of a region are used to compute the empirical mean $\hat{\boldsymbol{\mu}}_w$ and covariance $\hat{\Sigma}_w$.

Furthermore, in (2), encoding all texture vectors in $\hat{X}$ to represent region R is highly redundant because the N windows *overlap* with each other. Thus, to obtain an efficient code of R that closely approximates its true entropy, we only need to code the *nonoverlapping* windows that can tile R as a grid.

Ideally, if R is a rectangular region of size $mw \times nw$, where m and n are positive integers, then clearly we can tile R with exactly $mn = \frac{N}{w^2}$ windows. So for coding the region R, (2) becomes:

$$L_{w,\varepsilon}(R) \doteq (\tfrac{D}{2} + \tfrac{N}{2w^2})\log_2\det(I+\tfrac{D}{\varepsilon^2}\hat{\Sigma}_w) + \tfrac{D}{2}\log_2(1+\tfrac{\|\hat{\boldsymbol{\mu}}_w\|^2}{\varepsilon^2}). \tag{4}$$

Real regions in natural images normally do not have such nice rectangular shapes. However, (4) remains a good approximation to the actual coding length of a region R with relatively smooth boundaries.[2]

[2]For a large region with a sufficiently smooth boundary, the number of boundary-crossing windows is significantly smaller than the number of those in the interior. For boundary-crossing windows, their average coding length is roughly proportional to the number pixels inside the region if the Gaussian distribution is sufficiently isotropic.

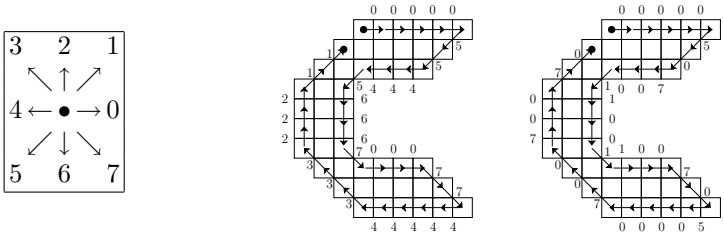


Figure 4: **Left:** The Freeman chain code of an edge orientation along 8 possible directions. **Middle:** Representation of the boundary of a region in an image w.r.t. the Freeman chain code. **Right:** Representation w.r.t the difference chain code.

2.3 Adaptive Boundary Encoding

To code windows from multiple regions in an image, one must know to which region each window belongs, so that each window can be decoded w.r.t. the correct codebook. For generic samples from multiple classes, one can estimate the distribution of each class label and then code the membership of the samples using a scheme that is asymptotically optimal for that class distribution (i.e., the Huffman code used in [13]). Such coding schemes are highly inefficient for natural image segmentation, as they do not leverage the spatial correlation of pixels in the same region. In fact, for our application, pixels from the same region form a connected component. Thus, the most efficient way of coding group membership for regions in images is to code the *boundary* of the region containing the pixels.

A well-known scheme for representing boundaries of image regions is the *Freeman chain code*. In this coding scheme, the orientation of an edge is quantized along 8 discrete directions, shown in Figure 4. Let $\{o_t\}_{t=1}^{T}$ denote the orientations of the T boundary edges of R. Since each chain code can be encoded using three bits, the coding length of the boundary of R is

$$B(R) = 3\sum_{i=0}^{7} \#(o_t = i). \tag{5}$$

The coding length $B(R)$ can be further improved by using an adaptive Huffman code that leverages the prior distribution of the chain codes. Though the distribution of chain codes is essentially uniform in most images, for regions with smooth boundaries, we expect that the orientations of consecutive edges are similar, and so consecutive chain codes will not differ by much. Given an initial orientation (expressed in chain code) o_t, the *difference chain code* of the following orientation o_{t+1} is $\Delta o_t \doteq \mod(o_t - o_{t+1}, 8)$. Figure 4 compares the original Freeman chain code with the difference chain code for representing the boundary of a region. Notice for this region, the difference encoding uses only half of the possible codes, with most being zeroes, while the Freeman encoding uses all eight chain codes. Given the prior distribution $P[\Delta o]$ of difference chain codes, $B(R)$ can be encoded more efficiently using a lossless Huffman coding scheme:

$$B(R) = -\sum_{i=0}^{7} \#(\Delta o_t = i) \log_2(P[\Delta o = i]). \tag{6}$$

For natural images, we estimate $P[\Delta o]$ using natural images from the BSD that were manually segmented by humans. We compare our distribution with one estimated by Liu and Zalik [20], who used 1000 images of curves, contour patterns and shapes obtained from the web. As the results in Table 1 show, the regions of natural images tend to have more smooth boundaries when segmented by humans.

Table 1: The prior probability of the difference chain codes estimated from the BSD and by Liu and Zalik [20].

Difference Code	0	1	2	3	4	5	6	7
Angle change	0°	45°	90°	135°	180°	−135°	−90°	−45°
Probability (BSD)	0.585	0.190	0.020	0.000	0.002	0.003	0.031	0.169
Probability (Liu-Zalik)	0.453	0.244	0.022	0.006	0.003	0.006	0.022	0.244

3 Image Segmentation Algorithm

In this section, we show how to use the coding length functions we developed in Section 2 to construct a better compression-based image segmentation algorithm. We describe the basic approach below, and then propose a hierarchical scheme to deal with small and/or thin regions.

3.1 Segmentation by Minimizing Coding Length

Suppose an image I can be segmented into non-overlapping regions $\mathcal{R} = \{R_1, \ldots, R_k\},\ \cup_{i=1}^{k} R_i = I$. The total coding length of the image I is

$$L^S_{w,\varepsilon}(\mathcal{R}) \doteq \sum_{i=1}^{k} L_{w,\varepsilon}(R_i) + \tfrac{1}{2} B(R_i). \tag{7}$$

Here, the boundary term is scaled by a half because we only need to represent the boundary between any two regions once. The optimal segmentation of I is the one that minimizes (7). Finding this optimal segmentation is, in general, a combinatorial task, but we can often do so using an *agglomerative* process.

To initialize the optimization process, one can assume each image pixel (and its windowed texture vector) belongs to an individual group of its own. However, this presents a problem that the maximal size of the texture window can only be one without intersecting with other adjacent regions (i.e., other neighboring pixels). In our implementation, similar to [13], we utilize an oversegmentation step to initialize the optimization by *superpixels*. A superpixel is a small region in the image that does not contain strong edges in its interior. Superpixels provide a coarser quantization of an image than the underlying pixels, while respecting strong edges between the adjacent homogeneous regions. There are several methods that can be used to obtain a superpixel initialization, including those of Mori et al. [21], Felzenszwalb and Huttenlocher [2], and Ren et al. [12]. We have compared the three methods in the experiment and found that [21][3] works well for our purposes.

Given an oversegmentation of the image, at each iteration, we find the pair of regions R_i and R_j that will maximally decrease (7) if merged:

$$(R_i^*, R_j^*) = \underset{R_i, R_j \in \mathcal{R}}{\operatorname{argmax}} \Delta L_{w,\varepsilon}(R_i, R_j), \quad \text{where}$$

$$\begin{aligned} \Delta L_{w,\varepsilon}(R_i, R_j) &\doteq L^S_{w,\varepsilon}(\mathcal{R}) - L^S_{w,\varepsilon}((\mathcal{R} \backslash \{R_i, R_j\}) \cup \{R_i \cup R_j\}) \\ &= L_{w,\varepsilon}(R_i) + L_{w,\varepsilon}(R_j) - L_{w,\varepsilon}(R_i \cup R_j) \\ &\quad + \tfrac{1}{2}(B(R_i) + B(R_j) - B(R_i \cup R_j)). \end{aligned} \tag{8}$$

$\Delta L_{w,\varepsilon}(R_i, R_j)$ essentially captures the difference in the lossy coding lengths of the texture regions R_i and R_j and their boundaries before and after the merging. If $\Delta L > 0$, we merge R_i^* and R_j^* into one region, and repeat this process, continuing until the coding length $L^S_{w,\varepsilon}(\mathcal{R})$ can not be further reduced.

To model the spatial locality of textures, we further construct a *region adjacency graph* (RAG): $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each vertex $v_i \in \mathcal{V}$ corresponds to region $R_i \in \mathcal{R}$, and the edge $e_{ij} \in \mathcal{E}$ is present if and only if regions R_i and R_j are adjacent in the image. To perform image segmentation, we simply apply a constrained version of the above agglomerative procedure – only merging regions that are adjacent in the image.

3.2 A Hierarchical Implementation

The above region-merging scheme is based on the assumption of a fixed texture window size, and clearly cannot effectively deal with regions or superpixels that are very small and/or thin. In such cases, the majority or all texture windows will intersect with the boundary of the regions. We say that a region R is *degenerate* w.r.t. window size w if $\mathcal{I}_w(R) = \emptyset$. For such a region, the w-neighborhoods of all pixels will contain pixels from other regions, and

[3] We use the publicly available code for this method available at `http://www.cs.sfu.ca/~mori/research/superpixels/` with parameter `N_sp` = 200.

so $\hat{\boldsymbol{\mu}}$ and $\hat{\Sigma}$ cannot be reliably estimated. These regions are degenerate precisely because of the window size; for any w-degenerate region R, there is $1 \leq w' < w$ such that $\mathcal{I}_{w'}(R) \neq \emptyset$. We say that R is *marginally nondegenerate* w.r.t. window size w if $\mathcal{I}_w(R) \neq \emptyset$ and $\mathcal{I}_{w+2}(R) = \emptyset$. To deal with these degenerate regions, we propose to use a hierarchy of window sizes. Starting from the largest window size, we recursively apply the above scheme with ever smaller window sizes till all degenerate regions have been merged with their adjacent ones. In this paper, we start from 7×7 and reduce to $5 \times 5, 3 \times 3$, and 1×1. For segmentation at smaller windows sizes, our scheme only allows adjacent regions to be merged if at least one of the regions is marginally nondegenerate.

Notice that at a fixed window size, the region-merging process is similar to the CTM approach proposed in [13]. Nevertheless, the new coding length function and the hierarchical implementation give much more accurate approximation to the true image entropy and hence lead to much better segmentation results (see Section 4). For completeness, we summarize the overall algorithm for image segmentation in Algorithm 1, which we refer to as *Texture and Boundary Encoding-based Segmentation* (TBES). On a Quad-Core Intel Xeon 2.5GHz machine, the superpixel initialization using the method of [21] takes roughly five minutes and our MATLAB implementation of Algorithm 1 takes approximately ten minutes per image.

Algorithm 1 (Texture and Boundary Encoding-based Segmentation)

Given image I, distortion ε, max window size w_M, superpixels $\mathcal{R} = \{R_1, \ldots, R_k\}$,

for $w = 1 : 2 : w_M$ **do**
  Construct $\hat{X}_w$ by stacking the $w \times w$ windows around each $p \in I$ as column vectors and applying PCA.
Construct RAG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} \simeq \mathcal{R}$ and $e_{ij} \in \mathcal{E}$ only if R_i and R_j are adjacent in I.
$w = w_M$
repeat
  if $w = w_M$ **then**
    Find R_i and R_j such that $e_{ij} \in \mathcal{E}$, $\mathcal{I}_w(R_i) \neq \emptyset$, $\mathcal{I}_w(R_j) \neq \emptyset$, and $\Delta L_{w,\varepsilon}(R_i, R_j)$ is maximal.
  else
    Find R_i and R_j such that $e_{ij} \in \mathcal{E}$, $\mathcal{I}_w(R_i) \neq \emptyset$, $\mathcal{I}_w(R_j) \neq \emptyset$, $\mathcal{I}_{w+2}(R_i) = \emptyset$ or $\mathcal{I}_{w+2}(R_j) = \emptyset$ and $\Delta L_{w,\varepsilon}(R_i, R_j)$ is maximal.
  if $\Delta L_{w,\varepsilon}(R_i, R_j) > 0$ **then**
    $\mathcal{R} := (\mathcal{R} \setminus \{R_i, R_j\}) \cup \{R_i \cup R_j\}$.
    Update $\mathcal{G}$ based on the newly merged region.
    $w = w_M$
  else if $w \neq 1$ **then**
    $w = w - 2$
until $\mathcal{I}_{w_M}(R) \neq \emptyset, \quad \forall R \in \mathcal{R} \quad$ **and** $\quad \Delta L_{w_M,\varepsilon}(R_i, R_j) \leq 0, \quad \forall R_i, R_j \in \mathcal{R}$
Output: The set of regions $\mathcal{R}$.

4 Experiments

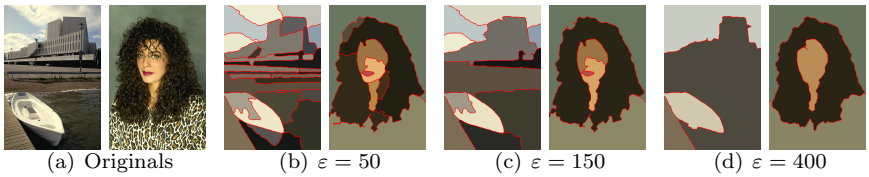
In this section, we conduct extensive evaluation to validate the performance of our method. We first describe our experimental setup, and then show both qualitative and quantitative results on two publicly available natural image databases (please refer to the supplemental material for a complete report of the segmentation results).

Experimental Setup. To obtain quantitative evaluation of the performance of our method we use three metrics for comparing pairs of image segmentations: the *probabilistic Rand index* (PRI) [22], the *variation of information* (VOI) [23], and the *precision* and *recall* of boundary pixels [5].[4] For brevity, we refer the reader to the stated references for the definition of each metric. In cases where we have multiple ground-truth segmentations, to compute a given metric for a test segmentation, we simply average the results of the metric between the test segmentation and each ground-truth segmentation. With multiple ground-truth segmentations for an image we can also estimate the human performance w.r.t. these metrics by treating each ground-truth segmentation as a test segmentation and computing the metrics w.r.t. the other ground-truth segmentations.

To apply Algorithm 1 to a natural image, we must choose the quantization level ε. As Figure 5 shows, a given image can have multiple plausible segmentations. We seek to find ε^* that tends to best match with human segmentation. To determine ε^* we run Algorithm 1 on each of the 100 test images in BSD for sixteen choices of ε

[4] We use the harmonic mean of precision and recall, known as the *global F-measure*, as a useful summary score for boundary precision and recall.

ranging from $\varepsilon = 25$ to $\varepsilon = 400$. We then choose ε that obtains the best average performance w.r.t. the various metrics. In our experiments we found that the choice of $\varepsilon^* = 150$ results in the best balance between the PRI and VOI metrics, so for all our subsequent experiments, we use this choice of ε.



(a) Originals (b) $\varepsilon = 50$ (c) $\varepsilon = 150$ (d) $\varepsilon = 400$

Figure 5: (color) Results of applying Algorithm 1 to two natural images for varying choices of ε.

Results on the Berkeley Segmentation Dataset. The Berkeley Segmentation Dataset consists of 300 natural images, each of which has been hand segmented by multiple human subjects. Figure 7 illustrates some representative segmentation results. We compare the performance of our method to five *publicly available* image segmentation methods, which we refer to as "CTM" [13], "MS" [3], "NC" [1], "UCM" [5], and "F&H" [2], respectively. Table 2 and Figure 6 summarize the performance of our method based on the various metrics for the BSD: the indices PRI and VOI in Table 2 are used to evaluate goodness of the regions; and the precision-recall curve and F-measure in Figure 6 evaluate the segmentation boundaries.

Table 2: Comparison of PRI and VOI for various algorithms on the BSD.

Index / Method	Human	TBES ($\varepsilon = 150$)	CTM	MS	NC	UCM	F&H
PRI (Higher is better)	0.87	**0.80**	0.76	0.78	0.75	0.77	0.77
VOI (Lower is better)	1.16	**1.76**	2.02	1.83	2.18	2.11	2.15

Notice that in Table 2, for both indices, our method achieves the best performance compared to all popular segmentation methods. It is also surprising that it does so with a fixed ε whereas CTM needs to rely on a heuristic adaptive scheme.

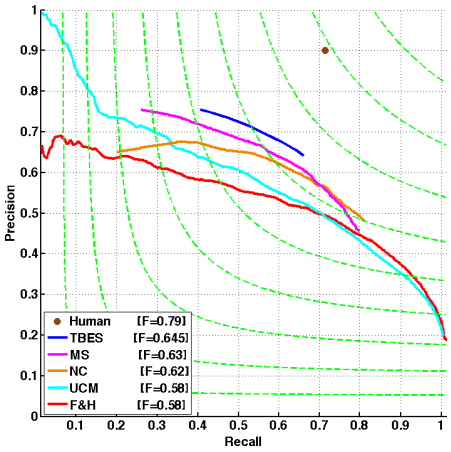


Figure 6: (color) Precision vs. Recall of boundaries on the BSD. The green lines are level sets of the F-measure, ranging from 0.1 (lower left) to 0.9 (upper right). Our method (TBES) is closest to human performance (brown dot), achieving an F-measure of 0.645.

For our segmentation results, if we could choose the best ε adaptively for each image to optimize the PRI index, the average PRI over the entire database would become 0.849; similarly for the VOI index, using the best ε for each image brings this index down to 1.466, both strikingly close to that of human segmentation. This suggests there is still plenty of room to improve our method by designing better schemes for choosing ε adaptively.

(a) Animals (b) Buildings (c) Homes (d) Nature (e) People

Figure 7: (color) Qualitative results of our algorithm on various kinds of images from BSD with a fixed $\varepsilon = 150$. For each result, the top is the original image, and the bottom is a segmentation image where each region is colored by its mean color.

Results on the MSRC Object Recognition Database. The MSRC Object Recognition Database consists of 591 images of objects grouped into 20 categories. We used the cleaned up segmentations provided by the authors of [24] as the ground truth. This dataset is highly challenging because images in MSRC are roughly 2/3 the size of those in the BSD, and in many cases, the ground truth segmentation only draws a boundary around the salient object in the image, casting everything else as background. Using our TBES algorithm with $\varepsilon = 150$, we obtained PRI =0.76, VOI =1.49, and F-measure = 0.53.[5] Compared to the BSD, our method did extremely well on MSRC in terms of VOI. The low F-measure is expected since usually only one region/object is segmented out on each image by a human. Some representative segmentation results of our algorithm are shown in Figure 8.

Figure 8: (color) Qualitative results of our algorithm on example images from MSRC with a fixed $\varepsilon = 150$. For each result, the top is the original image, and the bottom is a segmentation image where each region is colored by its mean color.

5 Conclusion

We have proposed a novel method for natural image segmentation. The algorithm uses a principled information-theoretic approach to combine cues of image texture and boundaries. In particular, the texture and boundary information of each texture region is encoded using a Gaussian distribution and adaptive chain code, respectively. The partitioning of the image is sought to achieve the maximum lossy compression using a hierarchy of window sizes. Our experiments have validated that this purely objective and simple criterion achieves state-of-the-art segmentation results on two publicly available image databases, both qualitatively and quantitatively.

Our proposed scheme for segmentation stimulates future investigation of its application for image compression. In this work, we rely on a closed-form formula for computing/estimating the coding length of a (Gaussian) distribution without explicitly computing a codebook. However, one can potentially use the proposed framework to explicitly build a codebook and compress the image. It will be interesting to explore how much better such a scheme performs against other popular lossy image compression methods such as JPEG and JPEG2000.

References

[1] Shi, J., Malik, J.: Normalized cuts and image segmentation. In: CVPR. (1997)

[2] Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. IJCV **59**(2) (2004)

[3] Comanicu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. PAMI **24** (2002) 603–619

[4] Elder, J., Zucker, S.: Computing contour closures. In: ECCV. (1996)

[5] Arbelaez, P.: Boundary extraction in natural images using ultrametric contour maps. In: POCV. (2006)

[6] Zhu, Q., Song, G., Shi, J.: Untangling cycles for contour grouping. In: ICCV. (2007)

[7] Ren, X., Fowlkes, C., Malik, J.: Learning probabilistic models for contour completion in natural images. IJCV (2008)

[8] Malik, J., Belongie, S., Leung, T., Shi, J.: Contour and texture analysis for image segmentation. IJCV (2001)

[5] Because we only have one ground truth segmentation per image in MSRC, we cannot compute the human performance w.r.t. the metrics.

[9] Tu, Z., Zhu, S.: Image segmentation by data-driven Markov Chain Monte Carlo. PAMI (2002)

[10] Kim, J., Fisher, J., Yezzi, A., Cetin, M., Willsky, A.: A nonparametric statistical method for image segmentation using information theory and curve evolution. PAMI (2005) 1486–1502

[11] Yu, S.: Segmentation induced by scale invariance. In: CVPR. (2005)

[12] Ren, X., Fowlkes, C., Malik, J.: Scale-invariant contour completion using condition random fields. ICCV (2005)

[13] Yang, A., Wright, J., Ma, Y., Sasty, S.: Unsupervised segmentation of natural images via lossy data compression. CVIU (2008)

[14] Ma, Y., Derksen, H., Hong, W., Wright, J.: Segmentation of multivariate mixed data via lossy coding and compression. IEEE TPAMI **29**(9) (2007) 1546–1562

[15] Martin, D., Fowlkes, C., Tal, D., Malik, J.: A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics. ICCV **2** (2001) 416–423

[16] Shotton, J., Winn, J., Rother, C., Criminisi, A.: Texton-boost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In: ECCV. (2006)

[17] Varma, M., Zisserman, A.: Texture classification: are filter banks necessary? In: CVPR. (2003)

[18] Levina, E., Bickel, P.J.: Texture synthesis and non-parametric resampling of random fields. Annals of Statistics **34**(4) (2006) 1751–1773

[19] Efros, A.A., Leung, T.K.: Texture synthesis by non-parametric sampling. ICCV (1999) 1033–1038

[20] Liu, Y.K., Zalik, B.: Efficient chain code with Huffman coding. Pattern Recognition **38** (2005) 553–557

[21] Mori, G., Ren, X., Efros, A., Malik, J.: Recovering human body configurations: combining segmentation and recognition. CVPR (2004)

[22] Rand, W.M.: Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association **66**(336) (1971) 846–850

[23] Meila, M.: Comparing clusterings: An axiomatic view. ICML (2005)

[24] Malisiewicz, T., Efros, A.: Improving spatial support for objects via multiple segmentations. BMVC (2007)