Robust Boundary Detection With Adaptive Grouping

Francisco J. Estrada
Centre for Vision Research, York University
Toronto, On., M3J 1P3, Canada
paco@elderlab.yorku.ca

Allan D. Jepson
Dept. of Comp. Sci., University of Toronto
Toronto, On., M5S 3G4, Canada
jepson@cs.utoronto.ca

Abstract

This paper presents a perceptual grouping algorithm that performs boundary extraction on natural images. Our grouping method maintains and updates a model of the appearance of the image regions on either side of a growing contour. This model is used to change grouping behaviour at run-time, so that, in addition to following the traditional Gestalt grouping principles of proximity and good continuation, the grouping procedure favours the path that best separates two visually distinct parts of the image. The resulting algorithm is computationally efficient and robust to clutter and texture. We present experimental results on natural images from the Berkeley Segmentation Database and compare our results to those obtained with three alternate grouping methods.

1. Introduction

Consider the following experiment: A human observer (perhaps the reader of this paper) is presented with a line drawing such as the one illustrated in Fig. 1 and asked to identify the central object in the image. After looking at it carefully for a moment, the observer might point out with remarkable confidence that he is looking at an image of a bear. In a surprisingly short interval, the observer has perceptually organized a large set of input features into meaningful groups and, at some point during the organization process, correctly identified the object in the image. A vast amount of research has been dedicated to identifying the principles that guide the formation of such groups and to developing algorithms that can exploit these principles to perceptually organize sets of input features.

In the particular case of line drawings, the above demonstration suggests that it should be possible to organize a set of input line segments using exclusively the information provided by the geometric relationships between them. Indeed, significant progress has been made in the fields of perceptual grouping and contour extraction by the clever exploitation of such geometric cues as proximity, smooth continuation, parallelism, and compactness. These cues, commonly referred to as Gestalt grouping principles or Gestalt laws after the school of psychologists that first investigated them at the turn of the 20th century [10, 20], have proven useful because they are good predictors of non-accidentalness [11]: groups of features that exhibit the above properties are unlikely to have originated randomly.

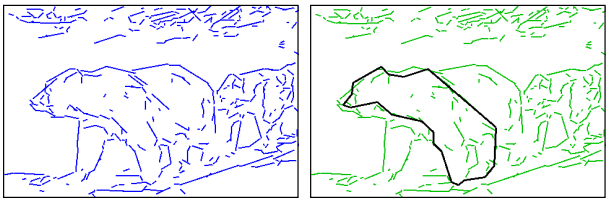

Figure 1. A complex line drawing with a bear inside it (left). Best contour detected with the grouping algorithm from [5] (right).

Gestalt principles are the foundation of all current contour extraction algorithms. However, our computational models of these principles are not yet sufficient to deal with complex line sets such as the one depicted in Fig. 1. In such images, the abundance of clutter and structured texture results in increased ambiguity that geometric cues may not be able to resolve, rendering contour grouping algorithms unreliable. The increased ambiguity also manifests itself as a dramatic increase in the computational expense required to perform grouping on complex line-sets. These problems have traditionally been alleviated by introducing domain specific knowledge, or by enforcing strong shape constraints such as convexity, but these approaches limit the applicability of grouping algorithms.

In this paper we show that an algorithm that adapts its grouping behaviour to the appearance of the image regions on either side of a growing contour is able to recover, reliably and robustly, the boundaries of objects in natural images with significant amounts of texture and clutter. Our method is based on the grouping algorithm of Estrada and Jepson [5]. Their algorithm provides an efficient search-based engine for contour extraction. We extend the algorithm to maintain and update a model of the appearance of

the image regions on either side of a growing contour. This model is used, in conjunction with traditional Gestalt cues, to guide the boundary extraction process so as to follow the path that best separates two visually distinct parts of the image. Once a complete contour has been found, the appearance model for the inside and outside of the contour can be used to measure the quality of the extracted boundary. The novel property of our algorithm is that it changes its grouping preferences at run-time depending on the characteristics of a particular grouping hypothesis. This allows the algorithm to accumulate information about the appearance of a potential object. Such information is particularly useful for finding the boundaries of heterogeneous objects, and is not available at the start of the search.

The main contributions of this paper are: 1) an adaptive grouping framework that uses both geometric information together with an evolving appearance model of potential objects; 2) a procedure for modifying the grouping behaviour of the algorithm at run-time, based on the appearance model for a particular (partial) contour hypothesis; 3) an appearance-based measure of contour quality used to rank detected shapes. 4) A quantitative evaluation and comparison of our algorithm and two alternate grouping methods on images from the Berkeley Segmentation Database (BSD). We also evaluate a reduced version of our framework that can use colour, but is not adaptive. We will show that the proposed algorithm is robust and efficient, and that it extracts boundaries that correspond more closely to individual object contours.

2. Previous Work

Because of space limitations, a comprehensive review of perceptual grouping research is not possible. Here we discuss only those methods that are directly related to the problem of detecting closed object contours from images. Mahamud et al. [12] propose an algorithm for contour extraction based on the stochastic completion fields of Williams and Jacobs [21]. The algorithm estimates the probability that a random walk started at one segment will visit other lines in the image and come back to the starting edge. These probabilities determine the saliency of segments and their connecting links. Closed contours are extracted as connected components derived from the edge and link saliencies. Elder and Zucker [2] describe a method that finds closed contours as shortest path cycles in a graph. Their formulation expands the representation of lines by including brightness values on either side of a segment. Elder et al. [1] extend this algorithm to extract closed boundaries by finding progressively longer chains of edges. Their algorithm explores only the edge sequences that have high likelihood with regard to a set of probabilities derived from natural scene statistics.

Jacobs [9] proposes an efficient, search based method for convex group extraction based on a threshold on boundary coverage (i.e. the percentage of a contour that is covered by image segments). Huttenlocher and Wayner [8] also use convexity as a constraint, but propose a greedy search algorithm. Saund [16] presents a grouping method that uses shape compactness (the ratio of a contour's area to the area of its convex hull), along with predefined preferences for particular grouping choices, to find closed boundaries on sketches. Estrada and Jepson [5] introduce an efficient search-based grouping algorithm based on local affinity normalization and shape compactness.

Sarkar and Soundararajan [18] use graph partitioning techniques to identify subsets of features that are weakly connected to the remaining features in a graph that encodes parallelism, perpendicularity, proximity, and continuity. Gdalyahu et al. [7] propose an algorithm that determines subsets of features that appear together on stochastically generated cuts in a graph. Wang et al. [19] present an algorithm that extracts closed contours as maximum likelihood cycles in a graph which encodes geometric relationships between neighboring edges. The likelihood of closed contours is normalized by the contour length to avoid a bias for small boundaries. They show that their Ratio Contour algorithm compares favorably to the grouping algorithms of Mahamud et al. [12], and Elder and Zucker [2].

A common problem for current contour grouping algorithms is that they rely almost exclusively on geometric relationships between edges, line segments, or small curve fragments. In general, current boundary extraction methods fail when confronted with images that are rich in clutter and structured texture. This is illustrated in Fig 1. Grouping in such images is extremely challenging because texture elements tend to align with object boundaries to yield a combinatorially large number of possible closed contours. In addition to this, clutter and texture combine to cause an explosive increase in grouping complexity that can render many grouping approaches impractical.

3. Robust Contour Extraction

Our search framework is based on that of Estrada and Jepson [5]. Their algorithm is capable of efficiently searching for closed contours in cluttered line-sets. For each pair of segments l_1 and l_2, our algorithm computes an affinity value given by

$$T_{\text{affty}}(l_1, l_2) = G_{\text{affty}}(l_1, l_2) + \gamma, \tag{1}$$

where the term $G_{\text{affty}}(l_1, l_2) \geq 0$ scores junctions according to geometric cues, and $\gamma > 0$ is a suitable constant. The form of this affinity function represents the belief that, on any given image, we will find two types of junctions: first, there are junctions that correspond to neighboring pairs of lines along object boundaries. Such junctions should be well-modeled by geometric grouping principles and should

receive a high $G_{\text{affty}}(l_1, l_2)$ score. Secondly, we have junctions that originate from un-structured texture, clutter, and noise. These junctions should receive a low $G_{\text{affty}}(l_1, l_2)$ score, and their affinity value will be dominated by the uniform term γ. The fact that such accidental junctions receive a similar affinity value (i.e. roughly γ) indicates that we have little reason to believe one of these junctions to be better than another.

The $G_{\text{affty}}(l_1, l_2)$ term we use here has a very simple form. It captures the geometric cues of proximity and smooth continuation, and has only one free parameter (notice that this is different from the form proposed in [5]). The proximity component is an exponential function of the gap between the segments measured as the distance d between their closest endpoints: $e^{-d^2/(2\sigma^2_{\text{gap}})}$ with an appropriate σ_{gap}. The smooth continuation term is proportional to the cosines of the angles between each segment and the line that joins the midpoints of these segments. If we call these angles θ_1 and θ_2 respectively, the smooth continuation term is given by $((\cos(\theta_1)+1)/2)*((cos(\theta_2)+1)/2)$. This term is 1 when the lines are collinear, and it decreases as the angles between the lines grow. The geometry used here is similar to that proposed by Elder and Zucker in [2], except that they measure θ_1 and θ_2 with regard to the line joining the closest endpoints of the two segments.

The G_{affty} score is calculated by multiplying together the proximity and smooth continuation terms. The affinity given by (1) is computed for every pair of lines in the image, but the grouping algorithm doesn't use the raw affinity scores. Instead, affinity values are locally normalized as follows: for each segment l_1 the algorithm finds the set $\mathcal{K}$ that contains the k segments with the largest affinities toward l_1. For each of these k segments, a normalized affinity score is computed as

$$N_{\text{affty}}(l_1, l_2) = \frac{T_{\text{affty}}(l_1, l_2)}{\sum_{l_i \in \mathcal{K}} T_{\text{affty}}(l_1, l_i)} \quad \in (0, 1).$$

The resulting normalized affinities sum to 1, and provide information about the *relative* goodness of possible grouping choices in the neighborhood of segment l_1. The properties of these normalized affinities are discussed in detail in [4], but it should be noted that they are not symmetric so in general $N_{\text{affty}}(l_i, l_j) \neq N_{\text{affty}}(l_j, l_i)$. The local normalization procedure leads to a robust and efficient search, and is particularly good at reducing search complexity in textured and cluttered regions of the line-set.

The contour grouping process is a depth-first search controlled by a threshold τ_{affty} on normalized affinities and by geometric constraints. In particular, each partial group is checked for simplicity (no self-intersections are allowed) and for compactness. That is, the ratio of the area of a contour to the area of its convex hull should be greater than some threshold τ_{compact}. The search algorithm is summarized as follows:

1 - For each segment i in the line-set, generate a group containing the single segment i.
2 - Find the set $\mathcal{K}$ with the best k junctions for the segment added most recently to the group (sorted by decreasing normalized affinity).
3 - For each segment j in $\mathcal{K}$

- If $N_{\text{affty}}(i, j) < \tau_{\text{affty}}$ end loop, otherwise add j to the group.
- Compute the compactness of the shape, if compactness $< \tau_{\text{compact}}$ try the next j.
- Check for closure, if the group is closed, compute its saliency and report it.
- Recursively call step 2 with the newly extended group.

4 - Report the extracted polygons sorted in order of decreasing saliency.

We will discuss how to estimate the saliency of a contour further on in the paper. Estrada and Jepson show that the use of normalized affinities leads to an efficient search even in the presence of clutter and un-structured texture. However, as illustrated in Fig 1, the presence of large amounts of structured texture leads their algorithm to form groups through textured regions of the image and miss the actual object contours. In what follows, we will describe the procedure that allows our algorithm to adaptively change the affinities between pairs of segments. We describe the appearance model used by the algorithm to judge region similarity, and discuss how this similarity can be used to change grouping behavior at run-time depending on a particular (partial) grouping hypothesis.

4. Building an Appearance Model for Partial Contours

There are many possible ways in which we could represent the appearance of the regions that lie on either side of a growing contour. We will not explore the important but separate problem of determining the optimal representation here. Instead we will use a simple, non-parametric representation that has been used extensively and with success in the context of image database retrieval [14]. We represent local appearance using colour histograms [17] in the $CIELab$ colour space [22]. The histogram representation is compact, easy to compute and manipulate, has well studied similarity measures [15], and, as we shall see, can be easily updated as additional knowledge about the appearance of an image region becomes available. In addition to this, the same model supports the use of other image cues such as texture [14].

Local histograms are computed within a small square window of fixed size w_{size} centered at a particular image location (x, y). Each histogram contains a fixed number n_{bins} of bins, and three layers corresponding to the three $CIELab$ colour components. The colour values at each pixel within the window are accumulated onto the

histogram with contributions weighted by a 2-D Gaussian mask (we ignore any pixel locations outside the image bounds). The resulting histograms are normalized to sum to 1. The choice of σ for this Gaussian is determined by the window size so that at least 1.5σ fit within the window on either side of (x, y). We will discuss in detail the appropriate choice of w_{size} and number of bins n_{bins} further on in the paper.

The local histograms need to be computed only once, before the search phase begins. Given the overlap between neighboring histograms, we could sample the image sparsely and still obtain a good representation. However, in the interest of simplicity, here we compute and store local colour histograms centered at every pixel in the image. To simplify notation in the next two sections of the paper, we will denote local histograms H as single layer, unit-area histograms that result from concatenating the individual histograms for the three layers of the $CIELab$ colour space.

4.1. Choosing the Optimal Histogram Characteristics

To determine the optimal values for w_{size} and n_{bins}, we performed an empirical evaluation of three histogram similarity measures on images from the Berkeley Segmentation Database. We randomly selected 75 images from the training image set of the BSD, and for each image we selected the human segmentation with fewest regions (under the assumption that this segmentation correctly captures the coarse structure of the image). We then randomly chose one of these 75 images, and extracted a pair of randomly located patches for a given w_{size}. We used the ground truth to determine whether the patches belonged to the same image region or not. We then computed histograms with different numbers of bins (between 2 and 50) from the extracted patches, and evaluated the similarity between them using histogram intersection, the χ^2 measure, and the JD-divergence (which is similar to the KL-divergence but has the advantage of being symmetric). See [15] for details on these measures.

We examined window sizes between 3 and 79 pixels, and for each window size we performed 2500 trials (each of them on a randomly chosen image). At the end of the sequence of trials, we computed a joint score for each combination of w_{size} and n_{bins}. The joint score is the average similarity for pairs of patches from the same image region plus the average dissimilarity for patches from different regions (where $\text{dissimilarity} = 1 - \text{similarity}$, or vice-versa if the chosen measure yields dissimilarity). The results are shown in Fig. 2. From these plots we can select the optimal values for w_{size} and n_{bins} that maximize the joint score. Notice that the three measures agree that smaller histograms with fewer bins yield the best results on the set of training images. Based on this, we set $w_{size} = 11$ and $n_{bins} = 4$ in our grouping method, and will use these values throughout our experiments.

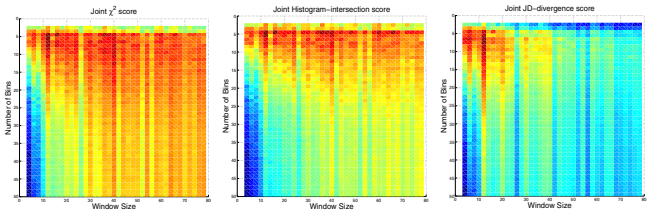

Figure 2. Joint score for the χ^2 (left), histogram intersection (centre), and JD-divergence (right) similarity measures for different window sizes and number of histogram bins. Blue indicates low scores, red indicates high scores.

To select the similarity measure, we follow the recommendations put forth by Rubner et al. in [15]. They performed a thorough experimental comparison of several measures of histogram similarity. Their results indicate that for the tasks of classification and image segmentation the χ^2 measure performs best at a reasonable computational cost (the Earth-Mover's distance has the best discrimination power, but Rubner et al. point out that for applications such as our algorithm that require the similarity measure to be evaluated a large number of times, its computational cost becomes prohibitive). For the above reasons, we have chosen to quantify histogram similarity using the χ^2 measure which for two histograms H_1 and H_2 is defined as

$$\chi^2(H_1, H_2) = \sum_i \frac{(f(i; H_1) - \hat{f}(i))^2}{\hat{f}(i)} \qquad (2)$$

where $f(i, H_1)$ is the value of the i^{th} bin of H_1, and $\hat{f}(i) = (f(i, H_1) + f(i, H_2))/2$. For unit-area histograms the χ^2 measure gives values in $[0, 1]$, where 0 indicates that the histograms are identical.

5. Adaptive Grouping

Given the colour histograms on either side of a partial contour, we can bias the growth of the contour so that it favours the path that best separates two dissimilar colour distributions. More formally, we say that each partial contour splits the image into two regions: an inside region, which lies to the right of the contour when traveling in the direction of contour growth, and an outside region that lies to the left of the contour. We maintain and update colour histograms for the inside and outside colour distributions along the partial contour.

Each segment to be considered for grouping is associated with two local histograms corresponding to the inside and outside (right and left respectively) with regard to the direction of contour growth (which always proceeds clockwise). To select the appropriate local histograms, we look in the direction perpendicular to the segment starting from its midpoint, and retrieve the pre-computed histograms at

a distance of $\pm(w_{size}+1)/2$. This choice ensures that we use histograms on either side of the segment that are close to the line, but still mostly contained within the inside or outside regions bounded by the contour. At the same time, the Gaussian weighting ensures that pixels close to the edges contribute little to the histograms, thus providing robustness to the colour variations that occur near image edges.

At search time, we compare the associated histograms of possible grouping choices with the colour histograms for the current contour, and we bias the search to favour grouping together edges that have similar colour distributions on the inside, as well as high dissimilarity between the inside and outside colour histograms. We denote the contour's inside histogram by $H_{\mathrm{b-in}}$, its outside histogram by $H_{\mathrm{b-out}}$, and the inside and outside histograms of a segment that is being considered as a possible extension to the contour by $H_{\mathrm{s-in}}$, and $H_{\mathrm{s-out}}$. We then use χ^2 to compute the similarity between the colour distributions on the inside, and the dissimilarities between the colour distributions on opposite sides of the boundary: $S_{in} = 1-\chi^2(H_{\mathrm{s-in}}, H_{\mathrm{b-in}})$, $D_{o\to i} = \chi^2(H_{\mathrm{s-out}}, H_{\mathrm{b-in}})$, and $D_{i\to o} = \chi^2(H_{\mathrm{s-in}}, H_{\mathrm{b-out}})$.

Here, S_{in} is simply the similarity between the inside colour distributions of the segment and the contour. $D_{o\to i}$ is the dissimilarity between the colour distribution on the outside of the new segment and the inside of the contour. It discourages grouping with segments that are inside an object (in which case, S_{in} is high, but $D_{o\to i}$ dissimilarity should be low). $D_{i\to o}$ is the dissimilarity between the inside of the new segment and the outside of the contour; it is useful when no candidate segment can be found whose inside colour distribution matches the inside of the contour. In this case, we favour the segment whose inside is most dissimilar to the outside of the current contour. Given these three terms, we define the histogram-based colour affinity between a segment l and a partial contour B as $C_{\mathrm{colour-affty}}(l,B) = (S_{in}+D_{o\to i}+D_{i\to o})/3$. This measure combines the three components of the colour affinity with equal weights. In general, an optimal weighting could be determined from results on training images.

Incorporating the colour affinity into the search procedure involves multiplying the original geometric affinity values by the newly computed colour affinities, and re-normalizing the resulting scores. For each line l_2 in the set $\mathcal{K}$ that contains the best grouping choices for the last segment l_1 in the current contour B, we compute

$$T_{\mathrm{affty}} = (G_{\mathrm{affty}}(l_1,l_2) * C_{\mathrm{colour-affty}}(B,l_2)) + \gamma,$$

and re-normalize the resulting, modified affinities. Once a segment has been chosen for grouping, it is added to the contour, and the segment's histograms are used to update the contour's appearance model. Since a contour with M segments has already accumulated M local histograms, the new inside histogram is given by $H_{\mathrm{b-in}} = ((M/(M+1))H_{\mathrm{b-in}})) + ((1/(M+1))H_{\mathrm{s-in}}))$, and similarly for the outside histogram. The search algorithm is identical to the one described on Section 3 and uses the same thresholds on normalized affinity and compactness.

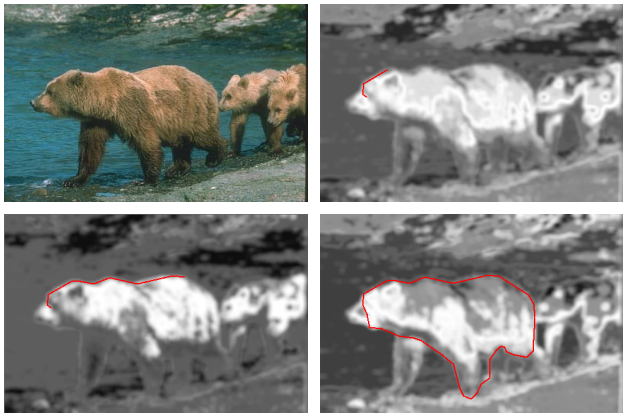

Figure 3. Clockwise from top-left: original image and three progressively longer contours (including the final, closed shape). For each contour, we show the similarity of the local histograms across the image with regard to the current model for the inside of the partial boundary (brighter is more similar). Notice that the similarity changes as the contour grows, and observe that parts of the image that should be inside in the bear are given high similarity.

The above procedure has several important effects: First, it changes the exploration order of potential grouping choices. The algorithm will favour segments whose colour histograms agree with the current appearance model for the partial contour. Second, it reduces the chance that the grouping algorithm will wander into textured regions even for structured texture patterns; this is because segments within a textured region are likely to have similar colour distributions on both sides. Third, it allows the algorithm to use information that was not available at the beginning of the search phase, and that keeps changing as the contour grows. The resulting adaptive search enables our algorithm to find object boundaries more accurately and with increased robustness. The adaptation process is illustrated in Fig 3. Notice that the model does a good job of estimating which parts of the image are likely to be part of the inside of the object bound by the partial contours.

6. Estimating The Quality of a Contour

The contour grouping phase is likely to find hundreds of closed groups on moderately complex line-sets. These have to be ranked so that only a few contours, deemed to be 'salient', are reported back to the user. In the current grouping literature, the saliency of groups is either determined as a side-effect of the optimization algorithm being used (i.e. the contour found first by a given optimization process is by the definition the most salient, like in [19]), or evaluated af-

terward using geometric information, either within a probabilistic (Bayesian) framework, such as in [5], or using especially defined saliency measures (for example, as in [12]). The important thing to notice is that such saliency measures depend almost exclusively on geometric information, and are thus likely to be unreliable in images with significant amounts of clutter and texture.

We propose that the quality of a contour should be related to the quality of the segmentation of the image induced by that contour. The best boundary neatly separates two visually distinct regions of the image. To quantify this, we evaluate the cost of encoding the image using the colour distributions inside and outside a hypothesized group. We compute colour histograms (here we will use again the original notation, with 3, unit-area colour layers per histogram) H_{in}, and H_{out} for the inside and outside of the contour respectively. Under the simplifying assumption that the three colour components for a given pixel are independent, the probability of drawing a pixel's colour $\vec{x} = (x_L, x_a, x_b)$ from a particular colour histogram is given by $p(\vec{x}|H) = H(B_{x_L}) \cdot H(B_{x_a}) \cdot H(B_{x_b})$, where $B_{x_{layer}}$ gives the appropriate layer and bin in the histogram for the indicated colour component of pixel x.

We estimate the encoding cost of a contour as the sum of the log-probabilities of drawing each pixel in the image from the appropriate region (inside or outside of the contour). For the shape S bound by the contour

$$EC = -\sum_{\vec{x}\in S} log(p(\vec{x}|H_{\text{in}})) - \sum_{\vec{x}\notin S} log(p(\vec{x}|H_{\text{out}})).$$

This measure favours boundaries that separate two different colour distributions because histograms for well-separated distributions have more probability mass on the bins of their corresponding pixels than histograms for more heterogeneous distributions (which are flatter, and get closer to uniform the more heterogeneous the corresponding region). All groups extracted by the algorithm are sorted, and the contours with the lowest encoding cost are presented at the top of the list.

7. Experimental Results

In this section we present a comparative study of grouping performance between our adaptive algorithm and three alternate grouping methods. The algorithms were tested on 21 images from the testing image set of the BSD. We selected only those images that contain at least one large, un-occluded object that is fully contained within the image bounds. We generated ground-truth object boundaries from the human segmentations provided with the BSD. Additionally, we selected 10 images from the training set of the BSD (following the same criteria) to be used as a training set for the parameters in our algorithm. The selected images depict complex objects in natural environments and usually contain large amounts of texture and clutter; as such, they constitute difficult perceptual grouping problems.

We compare our adaptive algorithm (AA) against a variation of our method that uses local colour histograms to compute a pairwise colour affinity for every pair of lines, but does not maintain or use an appearance model for the inside and outside of partial contours. The pairwise colour affinity is computed before the search phase begins and is multiplied directly onto the geometric affinity for the corresponding pair of segments. The resulting non-adaptive algorithm (NA) provides a baseline for judging whether the adaptation process provides any advantage over a method that uses colour, but not adaptation. We also compare AA against the original grouping method of Estrada and Jepson (EJ) [5], and against the Ratio Contour algorithm (RC) of Wang et al. [19][1].

All algorithms receive as input the same edge maps computed using the Canny edge detector. A robust line-fitting procedure was used to generate the line-sets for AA, NA, and EJ. Ratio Contour uses its own feature fitting process to generate a set of input fragments (small segments modeled with spline curves). The values of the parameters for our method were set on the training images. We discussed the optimal values for w_{size} and n_{bins} in the text above. In addition to these, our algorithm requires σ_{gap} and γ for the geometric affinity, and a compactness threshold τ_{compact}. We selected a low compactness threshold of $\tau_{\text{compact}} = .5$ to allow for reasonably non-compact shapes typical of objects such as long-legged animals (in contrast, the original EJ used $\tau_{\text{compact}} = .8$), and we set the values of $\sigma_{\text{gap}} = 15$ and $\gamma = .1$ so that the distribution of normalized affinities on our training images had the appropriate shape described in [4]. Finally, Estrada and Jepson suggest that values of τ_{affty} in $[1.1/K, 1.5/K]$ are the most useful, and we have observed that on our training images $\tau_{\text{affty}} = (1.1/K) = .055$ yields the best results. We use these same parameter values throughout all of our tests for both AA and NA. For EJ and RC we used the original parameters set by the authors of the corresponding algorithms, except that for EJ we used the same $\tau_{\text{affty}} = .055$ as with AA.

To evaluate the quality of a contour, we use a simple error measure based on the average distance between boundary pixels from two contours. If B_1 is a detected boundary, and B_2 is the corresponding ground truth for the image, we define the distance error between the two contours as

$$D_{error} = \frac{\sum_{i=1}^{N_{B_1}} dist(B_1(i), B_2)}{N_{B_1}} + \frac{\sum_{j=1}^{N_{B_2}} dist(B_2(j), B_1)}{N_{B_2}}$$

where N_{B_1} is the number of boundary pixels in B_1, $dist(B_1(i), B_2)$ is the distance between boundary point i in

[1] We would like to acknowledge the kindness of Dr. Song Wang in providing us with his implementation of RC.

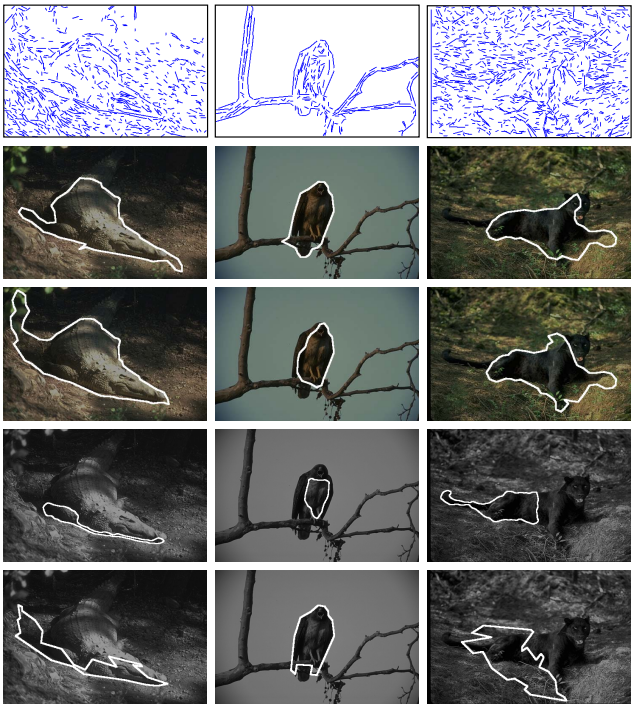

Figure 4. Results on 3 test images from the BSD (481×321 pixels in size). From top to bottom: input line-sets, best contours extracted by AA, best boundaries from NA, best RC contours (using RC's own segment fitting procedure), and best groups found by EJ. Note that RC and EJ do not use colour information.

B_1 and the closest boundary point in B_2; and similarly for N_{B_2} and $dist(B_2(i), B_1)$. This distance error will be small when the detected and ground-truth boundaries are aligned with each other without additional or missing sections.

For each image, we have a set of ground truth contours (corresponding to the different human segmentations of the same image and any large objects within it). For a given detected contour B_1, we compute and report the average D_{error} with regard to all available, overlapping ground-truth contours from the corresponding image. Each algorithm was allowed to generate 10 contours (using its own ranking method), and from these contours we selected the one with lowest distance error for comparison. We do this because otherwise we would be unable to determine whether any observed differences in performance are due to the algorithms themselves, or if instead they are simply the result of differences in the ranking of output shapes. Figure 4 shows the resulting contours for several challenging images from our testing set. Notice the complexity of the input and the abundance of texture and clutter.

The results show that the adaptive algorithm (AA) is able to retrieve contours that more closely approximate the boundary of the main object in each scene. It is also clear that the non-adaptive NA method, while better than the original EJ and RC, cannot match the quality of the contours extracted with AA. This indicates that the appearance model is in fact providing AA with the necessary guidance that enables it to detect good object boundaries. RC performs well, though it still has problems on very cluttered scenes. It appears to be the case that RC benefits from the smoothing introduced by the spline approximation used for fitting the input fragments. As a result, contours produced by RC are usually very smooth. EJ suffers from the complexity of the input and often yields groups that incorporate texture elements. Figure 5 shows the distance error obtained by each algorithm on each of the test images, as well as the average distance errors over the complete test set. This figure shows that the adaptive algorithm produces better results on most of the test images, achieving a significant reduction in boundary localization error. It is worth noting that the adaptive algorithm performs well on difficult images on which the competing methods perform poorly.

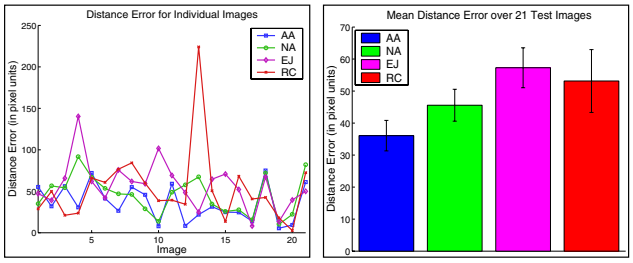

Figure 5. Left: distance error (as defined in the text) for all of the 21 test images. Right: mean distance error over the complete data set for each of the algorithms.

The average run-times for the algorithms are: 3.2 min. for AA, 4.2 min. for NA, 5.2 min. for RC, and .5 min. for EJ. It is interesting to see that NA takes longer to find contours. Since both algorithms require an equivalent amount of computation per search step, this indicates that the appearance model in AA not only produces better contours, but also reduces overall search complexity when compared to a purely local approach.

We note that our method has some limitations. The algorithm can fail to detect a reasonable boundary if an object and its background have similar colour distributions (e.g. if the main object is camouflaged), or if several objects with similar appearance are in close proximity. Also, algorithm parameters are scale-dependent. In particular, for images that are significantly larger than those used here, texture features on an object may become larger than the local-histogram windows; the resulting colour bias would favour the formation of groups around individual texture elements. Clearly, a multi-scale approach would be desirable. One of the authors is currently working on a multi-scale grouping method and preliminary results [3] appear promising. This multi-scale algorithm is not based on the grouping framework described here, but instead builds on the work of El-

der et al. [1]. On images half the resolution of those used here, the multi-scale method achieves a mean distance error (after adjusting for image size) of 36.32 which is very close to the error achieved by our adaptive algorithm. Results at full-resolution are worse due to problems with the current multi-scale implementation.

It is worth noting that humans achieve a mean distance error of just a couple of pixels which is much better than all of the grouping methods tested here. The gap in performance is similar to that observed for edge detection [13], and for image segmentation [6]. Clearly, some component of these tasks is not properly captured by current bottom-up algorithms. This should be a topic of reflection and discussion for the vision community.

The framework presented here could be extended without much effort to use texture information in the form of histograms of filter-bank responses (see [14]). The appropriate way to combine colour and texture can be learned from training images, as has previously been demonstrated by Martin et al. [13]. A more interesting extension involves the use of histograms of optical flow vectors to perform enhanced boundary extraction on motion sequences. These extensions are the topic of our current research. A more ambitious project would be to explore the use of recognition algorithms to modify the behaviour of the grouping phase at run-time. A match between part of an object boundary and a partial contour could be used to generate a spatial prior to guide contour formation. The grouping algorithm, in turn, could inform the recognition stage of its success or failure at extending a contour along some hypothesized shape, this could be used to either reject or refine possible object matches. However, robustly matching partial contours to object boundaries is a difficult and as yet open problem.

8. Conclusion

The method we have demonstrated here expands on current research in perceptual grouping by demonstrating that a grouping algorithm can efficiently and successfully change its behaviour at run-time in a context-dependent way. We show that an appearance model of the image regions on either side of a hypothesized contour can be used to guide boundary formation, and that dynamically adjusting grouping behaviour leads to the detection of better object boundaries on challenging natural images. Our method is efficient and robust, and it outperforms competing algorithms on complex images from the BSD.

References

[1] Elder, J., Krupnik, A., and Johnston, L., "Contour Grouping with Prior Models," *PAMI*, Vol. 25, No. 6, pp. 661-674, 2003. 2, 8

[2] Elder, J., and Zucker, S., "Computing Contour Closure," *ECCV*, pp. 399-412. 1996. 2, 3

[3] Estrada, F., and Elder, J., "Multi-Scale Contour Extraction Based on Natural Image Statistics,' *5th IEEE Workshop on Perceptual Organization in Computer Vision*, 2006. 7

[4] Estrada, F., and Jepson, A., "Controlling the Search for Convex Groups," *Tech-Report CSRG-842*, Department of Computer Science, University of Toronto, 2004. 3, 6

[5] Estrada, F., and Jepson, A., "Perceptual Grouping for Contour Extraction," *ICPR*, Vol. 2, pp. 32-35, 2004. 1, 2, 3, 6

[6] Estrada, F., and Jepson, A., "Quantitative Evaluation of a Novel Image Segmentation Algorithm," *CVPR*, Vol. 2, pp. 1132-1139, 2005. 8

[7] Gdalyahu, Y., Weinshall, D., and Werman, M., "Self-Organization in Vision: Stochastic Clustering for Image Segmentation, Perceptual Grouping, and Image Database Organization," *PAMI*, Vol. 23, No. 10, pp. 1053-1074, 2001. 2

[8] Huttenlocher, D., and Wayner, P., "Finding Convex Edge Groupings in an Image," *IJCV*, Vol. 8, No. 1, pp. 7-27, 1992. 2

[9] Jacobs, D., "Robust and Efficient Detection of Salient Convex Groups," *PAMI*, Vol. 18, No. 1, pp. 23-27, 1996. 2

[10] Koffka, K., *Principles of Gestalt Psychology*, Harcourt, Brace, & World, 1935. 1

[11] Lowe, D., "Three-Dimensional Object Recognition from Single Two-Dimensional Images," *AI*, Vol. 31, No. 3, pp. 355-395, 1987. 1

[12] Mahamud, S., Williams, L., Thornber, K., and Xu, K., "Segmentation of Multiple Salient Closed Contours from Real Images," *PAMI*, Vol. 24, No. 4, pp. 433-444, 2003. 2, 6

[13] Martin, D., Fowlkes, C., and Malik, J., "Learning to Detect Natural Image Boundaries Using Brightness and Texture," *NIPS*, 2002. 8

[14] Puzicha, J., Hofmann, T., and Buhmann, J. M., "Non-parametric Similarity Measures for Unsupervised Texture Segmentation and Image Retrieval," *CVPR*, pp. 267-272, 1997. 3, 8

[15] Rubner, Y., Puzicha, J., Tomasi, C., and Buhmann, J. M., "Empirical Evaluation of Dissimilarity Measures for Color and Texture," *CVIU*, Vol. 84, pp. 25-43, 2001. 3, 4

[16] Saund, E., "Finding Perceptually Closed Paths in Sketches and Drawings," *PAMI*, Vol. 25, No. 4, pp. 475-491, 2003. 2

[17] Swain, M., and Ballard, D., "Color Indexing," *IJCV*, Vol. 7, No. 1, pp. 11-32, 1991. 3

[18] Sarkar, S., and Soundararajan, P., "Supervised Learning of Large Perceptual Organization: Graph Spectral Partitioning and Learning Automata," *PAMI*, Vol. 22, No. 5, pp. 504-525, 2000. 2

[19] Wang, S., Kubota, T., Siskind, J. M., and Wang, J. "Salient Closed Boundary Extraction with Ratio Contour," *PAMI*, Vol. 27, No. 4, pp. 546-561, 2005 2, 5, 6

[20] Wertheimer, M., *A Sourcebook of Gestalt Psychology*, Routledge and Kegan Paul, 1938. 1

[21] Williams, L., and Jacobs, D., "Stochastic Completion Fields: A Neural Model of Illusory Contour Shape and Salience," *Neural Computation*, Vol. 9, No. 4, pp. 837-858, 1997. 2

[22] Wyszecki, G., and Stiles, W.S., "Color Science: Concepts and Methods, Quantitative Data and Formulae," Wiley, N.Y., 1982. 3