Scene Shape Priors for Superpixel Segmentation

Alastair P. Moore Simon J. D. Prince* Jonathan Warrell* Umar Mohammed Graham Jones†

University College London
Gower Street, WC1E 6BT, UK
{a.moore,s.prince}@cs.ucl.ac.uk

† Sharp Laboratories Europe
Oxford, OX4 4GB, UK
graham.jones@sharp.co.uk

Abstract

Unsupervised over-segmentation of an image into superpixels is a common preprocessing step for image parsing algorithms. Superpixels are used as both regions of support for feature vectors and as a starting point for the final segmentation. In this paper we investigate incorporating a priori information into superpixel segmentations. We learn a probabilistic model that describes the spatial density of the object boundaries in the image. We then describe an over-segmentation algorithm that partitions this density roughly equally between superpixels whilst still attempting to capture local object boundaries. We demonstrate this approach using road scenes where objects in the center of the image tend to be more distant and smaller than those at the edge. We show that our algorithm successfully learns this foveated spatial distribution and can exploit this knowledge to improve the segmentation. Lastly, we introduce a new metric for evaluating vision labeling problems. We measure performance on a challenging real-world dataset and illustrate the limitations of conventional evaluation metrics.

1. Introduction

Segmentation is a well studied problem and there has been much recent progress in supervised/user guided algorithms. However a solution to achieving good unsupervised segmentation remains elusive and performance on real-world datasets is low [5]. This is in part due to the fact that it is difficult to separate the processes of segmentation, detection and recognition: In the absence of simple metrics of homogeneity it is impossible to segment an object in a scene without having first located it (detection) and estimated some of its properties (recognition). While some effort has been made on combining these separate processes [11, 21] the variations that result from occlusion and changes of pose and lighting make this a challenging task.

Despite this difficulty recent progress has been made by oversegmenting the image. Small regions of the image, or *superpixels*, can serve a dual purpose: first, they act as a region of support for a feature vector. Second, classifying only superpixels reduces the degrees of freedom of the image model and facilitates efficient inference. There have been several different approaches using superpixels as a preprocessing step in state-of-the-art-vision algorithms: Firstly, in a hierarchy. Here a segmentation consists of a dendrogram and the relationship between segments can be used to facilitate recognition [11]. Secondly, as a set of hypotheses. Multiple segmentations can be used to find segments that are robust for detection and recognition [8, 18, 11]. Thirdly, as a sampling scheme. A fixed set of regions are used to label the image [15, 7].

The use of oversegmentation as a preprocessing step remains a bottom up process (although it may be subject to revision, see [10]). However, other closely related areas such as object detection and recognition have successfully exploited prior information. Examples include: object priors [20, 9] and scene category priors [7, 13].

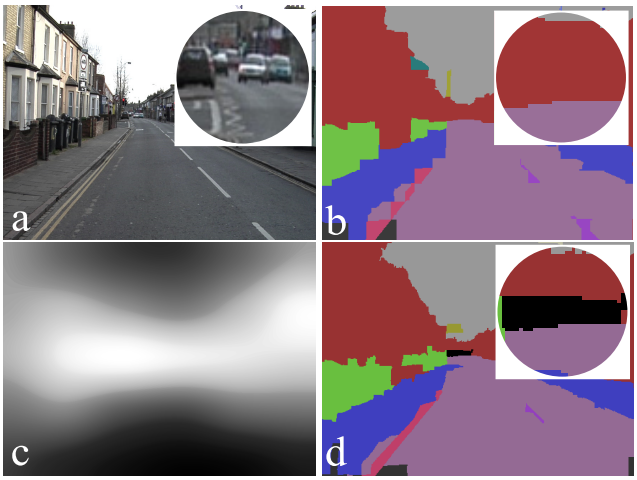

Figure 1. Scene shape priors. a) Street scene. Inset: cars in the distance. b) Segmentation without boundary prior. Inset: Small distant objects (cars) missed. c) Learned boundary distribution prior. d) Foveated segmentation based on learned prior. Inset: improved segmentation of distant objects.

*We acknowledge the support of the EPSRC Grant No. EP/E013309/1.

2009 IEEE 12th International Conference on Computer Vision (ICCV)
978-1-4244-4419-9/09/$25.00 ©2009 IEEE

In contrast there has been very little work on incorporating priors in the process of oversegmentation. For instance while it is noted in [6] that it is possible to have the segmentation method prefer components of certain size or shape this is not exploited. In [15] uniform size superpixels are encouraged using postprocessing regardless of where they appear in the image. Similarly, the algorithm presented in [14] implicitly assumes scenes that are largely frontoplanar or that objects of interest are reasonably large in the image. Alternatively there has been work placing priors on superpixels [8, 12] without it guiding the segmentation process itself. It remains to be seen whether it is possible to achieve good recognition rates without performing segmentation but in this paper we advocate an approach that uses priors in the process of oversegmentation. An example of one such spatial prior is shown in Figure 1.

The contributions of the paper are as follows: In Section 3 we describe an algorithm that uses an estimate of the distribution of boundaries in an image for foveation of the segmentation. In Section 4 we learn a model for the distribution of boundaries in an image from training data. In Section 5 we introduce a new metric for evaluating segmentation algorithms.

2. Superpixel Lattices

We begin by describing an existing superpixel lattice algorithm [14] and highlight its limitations. This motivates the use of priors in over-segmentation. In Section 3 we adapt this method to incorporate such prior information.

The superpixel lattice algorithm [14] divides the image into a regular grid of superpixels. The input to the algorithm is a *boundary map*. This is a 2D array containing a measure of the probability that a semantically meaningful boundary is present between two pixels. The boundary map is then inverted to take a value of 0 where there is the most evidence for a boundary and 1 where there is no evidence. This inverted map is called the *boundary cost map*.

The construction of the superpixel lattice is incremental: At each iteration the image is bi-partitioned in a vertical or horizontal strip. The goal is to segment the image in places where the *boundary cost map* is lowest, which is achieved by finding a minimum weighted path through each strip. This is illustrated in Figure 2 where four paths divide the image into a total of nine superpixels. After each iteration the weights along the chosen path in the *boundary cost map* are updated with a large fixed cost. This prevents parallel paths crossing and guarantees perpendicular paths cross only once. In addition the weights in a band surrounding each path are also increased to limit the minimum size of superpixels and to prevent subsequent paths following the same real-world boundary.

Although this algorithm has been shown to perform quite well, it implicitly imposes a quasi-uniform segmentation on

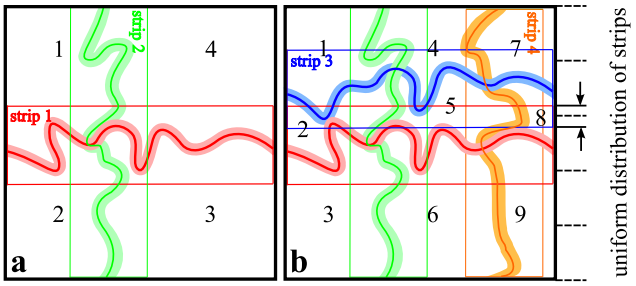

Figure 2. Incremental construction of superpixel lattice. a) The image is initially split left to right and top to bottom to form four regions. In each case we seek the optimal path within a predefined image strip. b) Adding one more vertical and horizontal path partitions the image into nine superpixels. Future path costs are modified in bands around previous paths (light colors) to prevent multiple-crossings and set a minimum distance of approach between paths. Strip overlap is marked with arrows.

the image by two mechanisms: Firstly, the strips are distributed evenly throughout the image leading to a roughly even distribution of paths. Second, the minimum cost path metric favors shorter paths across the image which are consequently relatively straight.

This quasi-uniform distribution is sensible if there is no *a priori* knowledge of the distribution of boundaries in the image. However, this is generally not the case. For many classes of image non-uniform sampling would be superior. For example, when a 2D image is a projection of a 3D scene, perspective effects result in an uneven distribution of the sizes of classes. In this paper, we segment road scenes where it is common for vehicles and pedestrians at a distance, near the vanishing point, to appear in small clusters in the center of the image. In this instance we would like foveation of the segmentation to take account of the expected difference in the sizes of classes.

3. An Adaptive Regular Lattice

Prior information about the spatial distribution of boundaries can be captured in the form of a *boundary distribution image*: each pixel takes a value between 0 and 1 representing the prior probability of observing a real-world boundary at this position (see Figure 3). The goal of this section is to show how to exploit the boundary distribution image to improve segmentation in the superpixel lattice algorithm. We first assume that we know this distribution and then learn this from training data in Section 4.

To exploit the boundary distribution prior in the superpixel lattice algorithm we must solve two problems: first we adapt the position and shape of the strips so that each has approximately the same prior probability of containing a boundary. This results in strips are no longer straight and vary in width across the image. Second, we adapt the minimum cost path algorithm so that the best path will tend to follow the shape of the strips.

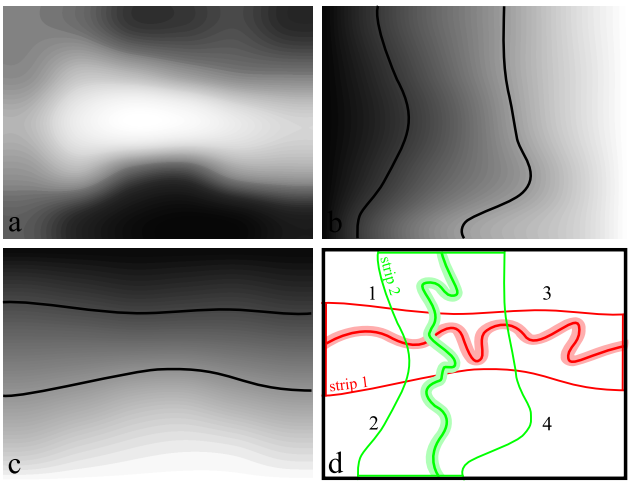

Figure 3. Construction of non-uniform strip a) Boundary distribution map representing probability of observing boundary at each position. b) Normalized cumulative distribution in horizontal direction. c) Normalized cumulative distribution in vertical direction. d) Boundaries of strips now follow iso-contours in these integral boundary maps. Compare with Figure 2a.

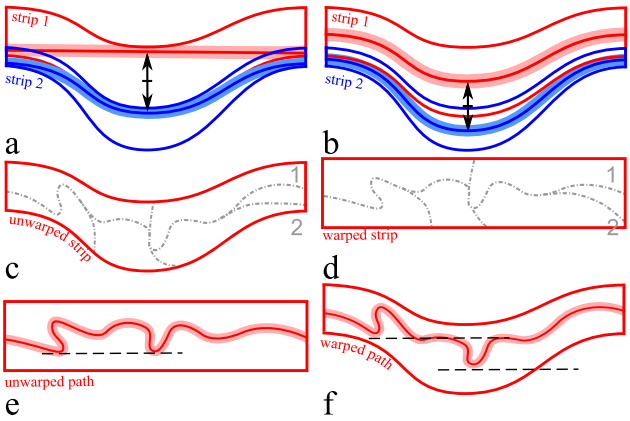

Figure 4. Warping minimum cost paths. a) Minimum cost path in unwarped strip. In the absence of boundary information paths go straight across strip. b) Minimum cost path in warped strip. Paths will follow the contour of the strip. Notice the difference in position and size of a superpixel, marked by an arrow, generated in warped and unwarped versions. c) Unwarped strip of boundary cost map. Note here that the shortest path takes branch 2 of the fork. d) Warped strip. In this strip the shortest path takes branch 1 of the fork. e) Minimum cost path in warped strip. d) Warped minimum path in unwarped strip.

3.1. Non-uniform Strips

We aim to calculate a set of strips across the image so that there is an equal chance of finding a boundary within each. We first discuss the assignment of vertical strips. For each row of the image we calculate the cumulative probability of observing a boundary as we move from left to right. This can be computed by integrating each row of the boundary distribution image and dividing by the total probability mass for that row. The result is a normalized cumulative distribution image (Figure 3b).

We now allocate strips so that they partition the cumulative distribution equally. In practice, this means that the edge of the strips follow the iso-contours of the normalized cumulative distribution image. For horizontal strips, we integrate the boundary distribution image in the vertical direction and normalize. We allocate strips that follow the iso-contours of this vertical normalized cumulative distribution. The result is a set of strips that are non-uniform - they may meander through the image and vary significantly in width depending on the position.

3.2. Non-Uniform Minimum Cost Path Algorithm

We now seek to find an optimal path for each strip that remains within the strip boundaries and bipartitions the image. The original superpixel lattice algorithm [14] used Dijkstra's algorithm to find the minimum cost path across the strip. The cost was determined by the boundary cost map (the complement of the probability of an boundary being present at each point). The algorithm hence tends to follow boundaries as these are cheap, but also aims to minimize path length. Unfortunately, this presents a problem with non-uniform strips. The path tends to take the straightest route across the image (Figure 4a) rather than follow the shape of the strip (Figure 4b). This effectively means that the superpixel size will not vary according to the prior.

To solve this problem we take the following approach: we warp the strip so that the average path along the warped version is now straight. We find the minimum cost path along the warped strip. Finally, we unwarp the path back to the original space. In this way we effectively define a distance metric along the strip so that shorter distances follow the shape of the boundaries (see Figure 4).

For a horizontal strip, the warping is achieved by applying a separate one dimensional affine warp to each column. The warps are chosen so that the strip boundaries in that column are always mapped to the same position. The inverse warp consists of applying the inverse one dimensional affine transform to each column.

By repeatedly finding non-uniform strips and finding optimal paths through these strips, it is possible to segment the image in such a way that it is influenced by the boundary distribution image. In particular, the superpixels will be smaller and more densely packed in regions where we expect to find more image boundaries.

4. Boundary Distribution Prior

Until now we have assumed that we know the boundary distribution image (BDI). In this section we describe an algorithm to take an observed set of images and infer the

most likely boundary distribution image. The image will depend on both prior information about the spatial statistics of boundaries (learnt from training data) and the observed edge data from a particular image. In this section we describe the model before describing learning (Section 4.1) and inference (Section 4.2) algorithms.

We wish to describe a probability distribution over observed boundaries $\mathbf{x} = [x_1 \dots x_P]^T$ at the p pixels of an image. Each element x_p is binary and is 1 when a boundary is present and 0 when it is absent. We assumes that x_p is drawn from a single observation of a binomial distribution with parameter y_p. Our goal then is to model the joint probability distribution of the vector of binomial parameters $\mathbf{y} = [y_1 \dots y_P]^T$. The vector $\mathbf{y}$ represents the boundary distribution image or BDI.

We assume that the distribution over $\mathbf{y}$ is determined by an underlying mixture of K clusters with each cluster having a subspace representation. We term this a "clustered latent trait" or CLT model [1]. More precisely, we assume that associated with image $\mathbf{y}_i$ there is (i) a discrete hidden variable c indicating which of K clusters generated the data and (ii) a continuous hidden variable $\mathbf{h}$ that represents the position within the subspace associated with that cluster. The variable $\mathbf{h}$ weights J basis functions $\mathbf{f}_{1k} \dots \mathbf{f}_{Jk}$ that form the columns of a matrix $\mathbf{F}_k = [\mathbf{f}_{1k} \dots \mathbf{f}_{Jk}]$ associated with the k'th cluster.

We define the activation $\mathbf{a} = [a_1 \dots a_P]^T$ for the i'th image to be a vector representing the propensity of each pixel to contain a boundary and calculate it as $\mathbf{a} = \mu_c + \mathbf{F}_c\mathbf{h}$ where μ_c is a mean vector that describes the average activation for cluster c. The activation $\mathbf{a}$ contains numbers defined on the whole real axis, and we convert these to a probability $\mathbf{y}$ by passing each element corresponding to pixel p through the logistic sigmoid function:

$$y_p = \sigma(a_p) = \frac{1}{1+\exp(-a_p)} \tag{1}$$

Finally, we assume that the probability of observing a boundary x_p at pixel p is given by y_p. We can summarize this model concisely as:

$$\begin{aligned} Pr(c=k) &= \pi_k && (2)\\ Pr(\mathbf{h}) &= \mathcal{G}_{\mathbf{h}}[\mathbf{0}, \mathbf{I}] && (3)\\ Pr(\mathbf{a}|\mathbf{h}) &= \delta_{\mathbf{a}}(\mu_c + \mathbf{F}_c\mathbf{h}) && (4)\\ Pr(\mathbf{x}|\mathbf{a}) &= \prod_{p=1}^{P} \text{Bin}_{\mathbf{x}_p}[\sigma(a_p)] && (5) \end{aligned}$$

where $\mathcal{G}_\alpha[\beta, \mathbf{\Gamma}]$ represents a Gaussian in variable α with mean β and covariance Γ. The function $\delta_\alpha(\beta)$ denotes a probability distribution over α where all of the mass is at β and hence describes a deterministic relationship. The function $\text{Bin}_\alpha[\beta]$ denotes the binomial likelihood of observing value α given binomial parameter β. The term π_k represents the prior probability of choosing the k'th cluster and in Equation 3 we have also defined a prior over $\mathbf{h}$. This graphical model is illustrated in Figure 5.

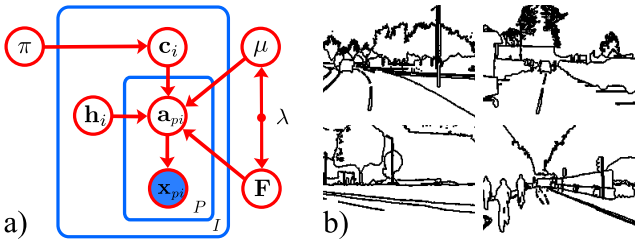

Figure 5. Graphical model. a) Clustered Latent Trait Model in which the plate denotes a set of I images each with P pixels. Here $\mu = \{\mu_k\}$, $F = \{F_k\}$. b) Example training data adapted from publicly available road sequences [2].

4.1. Learning

Our goal is to learn the parameters $\theta = \{\pi_{1\dots k}, \mu_{1\dots k}, \mathbf{F}_{1\dots k}\}$ of the CLT model based on I binary training images $\mathbf{x}_{1\dots I}$ where the value at each pixel represents the presence of a boundary. In particular we will maximize the joint log likelihood of all of the variables

$$\begin{aligned} L = \sum_{i=1}^{I} &[\log Pr(\mathbf{x}_i|\mathbf{h}_i, c_i, \mathbf{F}_{1\dots K}, \mu_{1\dots K}) + \log Pr(\mathbf{h}_i) \\ &+ \log Pr(c_i)] + \sum_{k=1}^{K} [\log Pr(\mathbf{F}_k) + \log Pr(\mu_k)] \end{aligned} \tag{6}$$

where we have defined Gaussian priors over the matrices of basis vectors $\mathbf{F}_{1\dots K}$ and the means $\mu_{1\dots K}$ so that

$$\begin{aligned} Pr(\mathbf{F}_k) &= \prod_{j=1}^{J} \mathcal{G}_{\mathbf{f}_{jk}}[\mathbf{0}, \lambda\mathbf{L}^{-1}] && (7)\\ Pr(\mu_k) &= \mathcal{G}_{\mu_k}[\mathbf{0}, \lambda\mathbf{L}^{-1}] && (8) \end{aligned}$$

where $\mathbf{f}_{jk}$ is the j'th column of matrix $\mathbf{F}_k$, $\mathbf{L}$ is the discrete approximation to the Laplacian operator and λ is a constant that was set by hand and controls the influence of the prior. These priors encourage spatial smoothness.

To describe the learning algorithm, first assume that the discrete variable $c_{1\dots I}$ representing the cluster assignments for each image are known. For each cluster, we use a Viterbi approach where we alternately maximize L with respect to the hidden variables $\mathbf{h}_i$ and the parameters θ. After several iterations, we reassign cluster assignments by finding the cluster $c_i \in \{1 \dots K\}$ under which the data $\mathbf{x}_i$ is most likely. This conditional likelihood is calculated using the optimal value of the hidden variable $\mathbf{h}_i$ for each cluster. We

Algorithm 1 Learn Clustered Latent Trait Model

for $t_1 = 1$ to T_1 **do**
  $\backslash\backslash$ *Reassign data to clusters*
  for all i **do**
    $c_i, \mathbf{h}_i \leftarrow \arg\max_{c_i, \mathbf{h}_i} \log Pr(\mathbf{x}_i, \mathbf{h}_i, c_i, \theta)$
  end for
  $\backslash\backslash$*Update parameters for each cluster*
  for $t_2 = 1$ to T_2 **do**
    for all k **do**
      $\theta_k \leftarrow \arg\max_{\theta_k} \sum_{s \in \{s:c_s=k\}} \log Pr(\mathbf{x}_s, \theta, c_s, \mathbf{h}_s)$
    end for
    for all i **do**
      $\mathbf{h}_i \leftarrow \arg\max_{\mathbf{h}_i} \log Pr(\mathbf{x}_i, \mathbf{h}_i, \theta, c_i,)$
    end for
  end for
end for
return $\theta_{1...K}$

also re-estimate the prior probability $\pi_{1...K}$ of each cluster. Having reassigned the points, we then relearn each cluster separately and so on. Algorithm 1 describes this process more formally. The maximization over the discrete parameters c_i was done exhaustively. The maximization over the continuous parameters θ and $\mathbf{h}_{1...I}$ was performed using a quasi-Newton method. The update for $\pi_1 \ldots \pi_k$ is calculated in closed form:

$$\pi_k = \frac{1}{I} \sum_{i=1}^{I} \delta(c_i = k) \tag{9}$$

Examples of the boundary distribution image $\mathbf{y}$ associated with each cluster mean are illustrated in Figure 6.

4.2. Inference

In this section we describe how to predict the boundary distribution image that was most likely to have been responsible for a new observed image. We calculate a binary edge map for the new image and use this as a proxy for the unseen boundary map $\mathbf{x}_t$. We then find the the cluster c_t and hidden variable $\mathbf{h_t}$ that were most likely to have created it:

$$c_t, \mathbf{h}_t \leftarrow \arg\max_{c_t, \mathbf{h}_t} \log Pr(\mathbf{x}_t, \mathbf{h}_t, c_t, \theta) \tag{10}$$

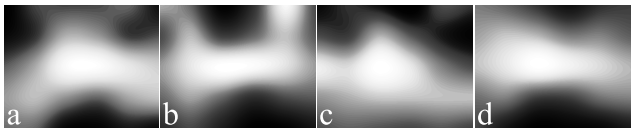

Figure 6. Learned cluster means. Each mean has a strong peak in the center of the image (a common boundary distribution for road scenes) but there are subtle changes in the distribution around this point. For instance the black 'sky' region shifts from the left of the image in b) to the right in c).

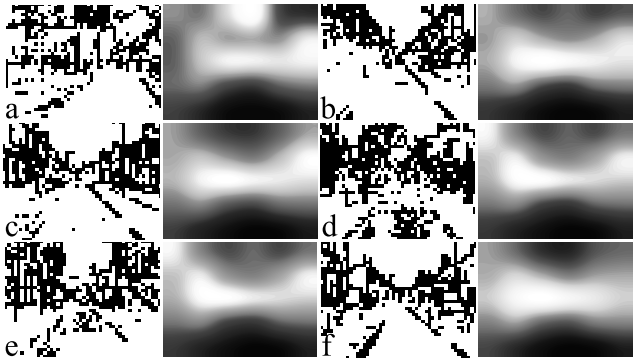

Figure 7. Sampling from one cluster during inference. Image pairs consist of output of a unary classifier, $\hat{\mathbf{x}}_\mathbf{t}$, and maximum likelihood $\mathbf{y_t}$. Notice the shift from right to left from a) to f) as the direction of the road changes suggesting the sub-space model is a useful representation.

We use the generative model to calculate the activation $\mathbf{a}_t = \mu_{\mathbf{c_t}} + \mathbf{F}_{c_t}\mathbf{h}_t$ associated with these variables. Finally, the elements of the binomial probability vector $\mathbf{y}_t$ are calculated using Equation 1. This process is illustrated in 7.

5. Evaluation Methodology

Before presenting results, we first introduce a new metric for evaluating superpixel algorithms. A frequently-used existing measure is pixel segmentation performance using precision ($P = \frac{TP}{TP+FP}$) and recall ($R = \frac{TP}{TP+FN}$). These standard measures are a conservative estimate of performance as neither is normalized by the true negative count. The *F-measure* [17]:

$$F = \frac{RP}{(\alpha R + (1-\alpha)P)} \tag{11}$$

is the harmonic mean that captures the trade off between true signal (Recall) and noise (Precision) for a segmentation, F_s, and in these experiments we set $\alpha = 0.5$.

A second possible performance metric is detection. This is the measure of how many separate instances of an object of a particular class are found in an image. We calculate object detections from our superpixel segmentation by merging together neighboring pixels where the modal ground truth class is the same (see Figure 8). We greedily associate segmented objects to the ground truth objects and define a correct detection (hit) when the bounding boxes of the segmented (B_p) and ground truth (B_{gt}) objects overlap by more than 50%. Misses constitute ground truth objects with no sufficiently overlapping segmented object. False detections constitute segmented regions that do not correspond or sufficiently overlap to ground truth regions. The overall quality of detection F_d is calculated by taking the harmonic mean of precision and recall using Equation 11.

Neither segmentation nor detection alone completely captures our ability to parse a scene (see Figure 9). We

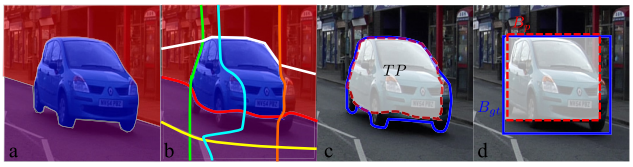

Figure 8. Detection with Oversegmentation. a) Ground truth data b) Segmentation with each superpixel labeled with mode class of ground truth data. c) Combined superpixels with true positive region highlighted. d) Bounding box overlap between ground truth and detected region.

therefore propose a new metric, that treats both segmentation and detection as independent measures of performance, and use the harmonic mean of the two separate *F-measures* as our summary statistic:

$$F_{sd} = \frac{2F_s F_d}{(F_s + F_d)} \tag{12}$$

Equation 12 will tend to 1 when the scene is parsed exactly. It takes into account the accuracy of the segmentation while penalizing those that split, merge or miss objects in the ground truth data[1].

6. Quantitative Results

Our evaluation is based on video sequence stills and human-labeled ground truth from the *CamVid* database [2]. This consists of road scene sequences taken from the passenger seat of a moving car. This is a challenging dataset that includes 32 classes and ego-motion. We follow [3] and focus on 11 of the most common classes shared between training (06R0,16E5) and test (05VD) sets. The *boundary cost map* is based on the output of the *boosted edge learning* algorithm (BEL) [4]. To learn the model presented in Section 4 we construct training data from the set of 406 binary ground truth boundary images by down-sampling from 720×960 to 36×48 using an **OR** operation. As the spatial prior only influences the distribution of strips, not the final minimum paths, it is possible to use very low resolution images. We learn the CLT model with 3 factors and either 1 or 4 clusters, set by hand. For inference with a new image we threshold the output of the BEL algorithm at 0.5, downsample and vectorize this binary image to form $\mathbf{x}_t$. We use a 20×25 lattice with a strip overlap of 0.49. In implementing the analysis described in Section 5 we ignore regions smaller than 10 pixels to mitigate some of the variability in the ground truth data.

Results using the CLT model with the adaptive lattice can be seen in rows 6-7 of Table 1. We can see that there is a modest improvement when using the clustered model rather than a single factor matrix but 75% of samples are drawn from one cluster. We note that the test data is less varied than the training data and the true benefit of using

[1]Code will be made publicly available from http://pvl.cs.ucl.ac.uk/

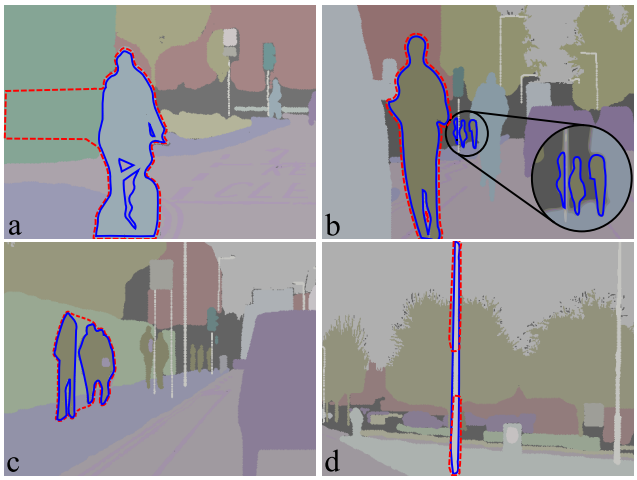

Figure 9. Segmentation vs. Detection. Four illustrative examples. a) Hit. Good detection but poor segmentation. b) Miss. Good segmentation but poor detection. c) Merge. Good segmentation but poor detection. d) Split. Good segmentation but poor detection.

the cluster model may be underestimated using this partition of the data. Similarly to [14] it is possible to relax the restriction of the fixed topology of the lattice by subsequently merging superpixels. We improve results of the CLT model for a fixed number of 500 superpixels by starting with a higher resolution grid 23×28 and then greedily merging based on the boundary cost map along the shared edge. When the merging order is tied, we prioritize merges in areas that the CLT model assigns a low boundary probability to. This usually results in sky and road being grouped before other regions. Results for the merged lattice can be seen in rows 8-9 of Table 1. Increasing the lattice resolution and merging pixels sacrifices the benefits of a regular grid, but improves the results and makes comparison with other algorithms that have no topological constraint more fair.

6.1. Comparison to other algorithms

To directly assess the effect of the learned spatial prior, we first compare our results to the original superpixel lattice algorithm [14]. Examples of segmentations with and without the prior are shown in Figure 11. In Figure 10a we can see a significant improvement when comparing the mean performance over test images. We can also see from Figure 10e that poor performance outliers are completely removed by incorporating the prior. While this is encouraging we can gain greater insight by evaluating the performance per class. In Table 1 we can see that the class average F_{sd} increases from 0.52 to 0.55. When gauging the sensitivity of this new metric it is worth noticing that simply downsampling the ground truth labeling by a factor of 2 results in an average 10% drop in performance. If we consider particular classes, rather than the average, there is a large improvement in classes that vary in size due to perspective

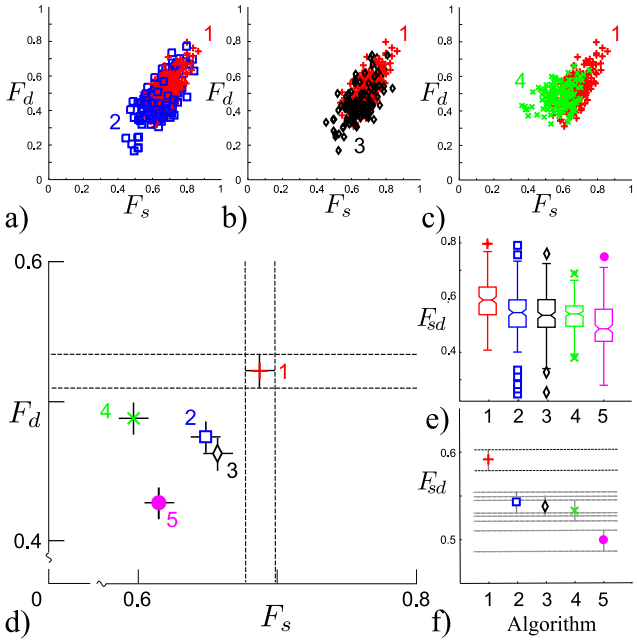

Figure 10. Plots of F_s, F_d and F_{sd} for algorithms 1-5. Test set contains 171 images. Key: [1+] Our best CLT model with 500 superpixels. [2□] 20×25 superpixel lattice [14]. [3◇] Implementation of normalized cuts using Pb boundary map with average 510 superpixels [15]. [4×] Minimum spanning tree with average 558 superpixels [6].[5•] 20×25 uniform sampling. a) Algorithm 1 vs. 2. b) Algorithm 1 vs. 3 c) Algorithm 1 vs. 4. d) Plot of F_s against F_d showing cluster means and standard errors. e) Box plot for each algorithm using combined F_{sd} score. f) Cluster means and standard error for F_{sd}. Our best method using a spatial prior significantly increases performance over superpixel algorithms without spatial prior.

effects with a $4\%, 8\%$ and 9% increase for bicyclist, car and pedestrian classes respectively.

We also compare our best method to other competing algorithms. Firstly, we compare to an implementation of normalized cuts [19] using the Pb boundary map with an average of 510 superpixels [15]. Due to memory constraints this was run on a down-sampled 360×480 image. The effect of this downsampling can be evaluated in row one of Table 1. In addition we compare to the minimum spanning tree algorithm [6] with an average of 558 superpixels. As regions smaller than 10 pixels are ignored from the analysis the small regions that this algorithm can generate do not degrade its measured performance. Using our analysis we can see from Figure 10d that the algorithm performs second best for detection but last for segmentation. This is highlighted again in Table 1 where it scores highest for smaller object categories Column/Pole and Sign/Symbol but has the poorest average score. It also scores very well for the Sky class which has reasonably uniform appearance. Finally, we include the result of simply down-sampling the image into a regular 20×25 grid of square superpixels.

From Figure 10f we see that there is no significant difference in performance amongst the competing algorithms. However, our best model outperforms all other algorithms and paired t-tests with the Bonferroni correction indicate that this difference is statistically significant ($p<0.05$). The final column of Table 1 shows that these differences are not clearly revealed if we only consider the global pixel accuracy metric used in [14].

7. Discussion and Conclusions

In this paper we have introduced a novel boundary density prior for superpixel algorithms. We have shown how to adapt an existing superpixel algorithm to take this density into account. We have also introduced a more sensitive metric for evaluating performance in image labeling problems and used this to demonstrate that the use of our prior improves segmentation performance.

Our study reveals the inadequacy of using simple pixel accuracy to assess over segmentation algorithms: we are using the ground truth data as a ideal classifier and therefore performance on individual pixels is very high for all methods. Results for our proposed metric suggest that despite this high accuracy there is plenty of room for improvement.

In this paper we have learnt a prior for the segmentation from real-world boundaries in training data. However, we note that other methods would be possible. For example, [9] estimated surface geometry and camera viewpoint to determine a prior over object sizes and a similar approach could be used to influence segmentation. Essentially, our model has learnt that road scenes tend to be foveated without high level information about the perspective projection process.

The use of low resolution priors for guiding segmentation also has interesting analogies to other work. While our method is based on the local unary classifier and therefore not directly comparable the title "scene shape" is motivated by the notions of "shape envelope" [16] used for scene recognition. Our results are achieved without any explicit object knowledge, object location priors or even class specific edges. In future work, we hope to explore the use of a temporal model and semi-supervised techniques for learning when the availability of labeled data is limited.

References

[1] D. Bartholomew and M. Knott. *Latent variable models and Factor analysis*. London, 1999.

[2] G. Brostow, J. Fauqueur, and R. Cipolla. Semantic obje classes in video: A high-definition ground truth databse. *Pattern Recognition Letters*, 2007.

[3] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla. Segmentation and recognition using structure from motion point clouds. pages 44–57, 2008.

[4] P. Dollr, Z. Tu, and S. Belongie. Supervised learning of edges and object boundaries. *CVPR*, 2:1964–1971, 2006.

Algorithms	Bicyclist	Building	Car	Column/Pole	Fence	Pedestrian	Road	Sidewalk	Sign/Symbol	Sky	Tree	Class Average	Accuracy
ground truth 360×480	0.95	0.92	0.97	0.80	0.97	0.95	0.88	0.97	0.96	0.84	0.71	0.90	0.99
uniform 20×25	0.26	0.63	0.54	0.06	0.68	0.31	0.60	0.75	0.24	0.66	0.58	0.48	0.88
FH $\sim$ 558 [6]	0.31	0.52	0.58	**0.35**	0.45	0.29	0.61	0.54	**0.46**	**0.84**	0.45	0.49	0.87
NC $\sim$ 510 [15]	0.37	0.65	0.65	0.10	0.71	0.34	0.62	0.76	0.30	0.70	0.61	0.53	0.90
GRL 20×25 [14]	0.33	0.64	0.63	0.23	0.70	0.28	0.63	0.77	0.28	0.71	0.59	0.52	0.90
CLT 1 Cluster 20 × 25	0.36	0.63	0.71	0.21	0.71	0.35	0.63	0.77	0.31	0.70	0.58	0.54	0.90
CLT 4 Clusters 20 × 25	0.37	0.63	0.71	0.22	0.73	0.37	0.65	0.80	0.31	0.70	0.61	0.55	0.90
CLT Merged 1 Clusters	0.36	0.65	**0.75**	0.26	0.73	0.39	0.63	0.79	0.36	0.69	0.60	0.56	0.90
CLT Merged 4 Clusters	**0.38**	0.64	0.74	0.23	**0.74**	**0.42**	0.63	0.79	0.39	0.69	0.60	**0.57**	0.90

Table 1. Results of F_{sd} by class for competing algorithms. Performance on classes that very in size with perspective increases dramatically with a conservative 8% and 9% improvement for pedestrian and cars respectively over competing methods.

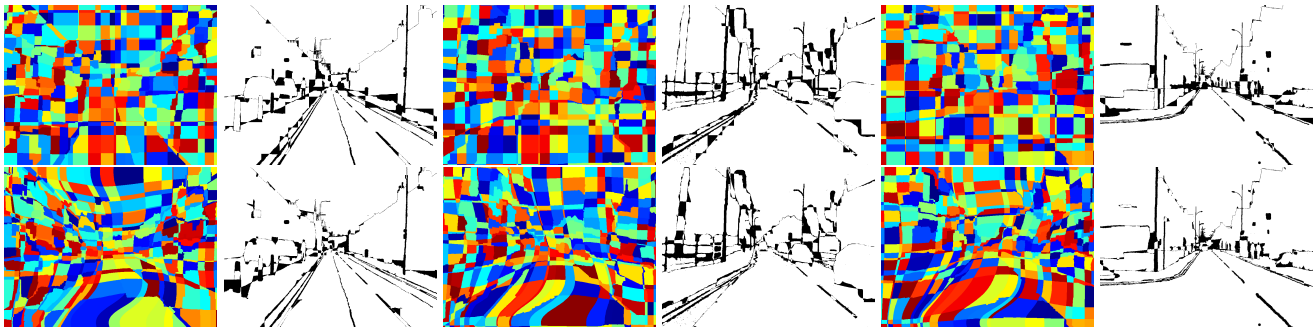

Figure 11. Best/Median/Worst results for our approach. Row 1, the superpixel lattice [14]. Row 2, our adaptive lattice using CLT prior. The effect of the learnt prior is a foveation of the superpixel segmentation. Image pairs consist of randomly colored superpixels with pixel errors in black. Notice also that the worst performing image has visibly few black regions but produces a lower score. This is because small objects on the pavement (signs/pedestrians) are split, missed or merged even if the pixel error remains small. This highlights the benefit of our analysis.

[5] M. Everingham, L. V. Gool, C. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge workshop. In *ECCV*, 2008.

[6] P. Felzenszwalb and D. Huttenlocher. Efficient Graph-Based Image Segmentation. *IJCV*, 2(59):167–181, 2004.

[7] X. He, R. S. Zemel, and D. Ray. Learning and incorporating top-down cues in image segmentation. *ECCV*, 1:338–351, 2006.

[8] D. Hoiem, A. Efros, and M. Hebert. Automatic photo pop-up. *ACM SIGGRAPH*, 2005.

[9] D. Hoiem, A. Efros, and M. Hebert. Putting objects in perspective. *CVPR*, 2006.

[10] D. Hoiem, A. Efros, and M. Hebert. Closing the loop on scene interpretation. *CVPR*, 2008.

[11] D. Hoiem, A. N. Stein, A. A. Efros, and M. Hebert. Recovering occlusion boundaries from a single image. *ICCV*, 1:1–8, 2007.

[12] P. Kohli, L. Ladicky, and P. Torr. Robust higher order potentials for enforcing label consistency. *CVPR*, 2008.

[13] L. Li and L. Fei-Fei. What, where and who? classifying event by scene and object recognition . *ICCV*, 2007.

[14] A. P. Moore, S. J. D. Prince, J. Warrell, U. Mohammed, and G. Jones. Superpixel lattices. *CVPR*, 2007.

[15] G. Mori. Guiding model search using segmentation. *ICCV*, 2:1417–1423, 2005.

[16] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42(3):145–175, 2001.

[17] C. V. Rijsbergen. *Information Retrieval*. Department of Computer Science, University of Glasgow, 1979.

[18] B. C. Russell, A. A. Efros, J. Sivic, W. T. Freeman, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. *CVPR*, pages 1605–1614, 2006.

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

[20] A. Torralba. Contextual priming for object detection. *IJCV*, 53(2):169–191, 2003.

[21] Z. Tu, X. Chen, A. Yuille, and S.-C. Zhu. Image parsing: Unifying segementation, detection and recognition. *IJCV*, 2(63):113–140, 2005.