Towards Fast and Accurate Segmentation

Camillo Jose Taylor
GRASP Laboratory
University of Pennsylvania, Philadelphia PA. USA
cjtaylor@cis.upenn.edu

Abstract

In this paper we explore approaches to accelerating segmentation and edge detection algorithms based on the gPb *framework. The paper characterizes the performance of a simple but effective edge detection scheme which can be computed rapidly and offers performance that is competitive with the pB detector. The paper also describes an approach for computing a reduced order normalized cut that captures the essential features of the original problem but can be computed in less than half a second on a standard computing platform.*

1. Introduction

Segmentation, the problem of breaking a given image into salient regions, is one of the most fundamental issues in Computer Vision and a number of approaches have been advanced to accomplish this task. Among these schemes, three methods have proven to be quite popular in practice owing to their performance and/or their running time. The Mean Shift method of Comaniciu and Meer [6], the Normalized Cuts method developed by Shi and Malik [14] and the graph based method of Felzenswalb and Huttenlocher [8].

More Recently Arbelaez, Maire, Fowlkes and Malik [3, 2] have proposed an impressive segmentation algorithm that achieves state of the art results on commonly available data sets. Their *gPb* method starts with a local edge extraction procedure which has been optimized using learning techniques. The results of this edge extraction step are then used as input to a spectral partitioning procedure which globalizes the results using Normalized Cuts. This globalization stage helps to focus attention on the most salient edges in the scene.

The globalization procedure is very effective but it involves the solution of a large, sparse eigensystem which can be quite time consuming. Published results indicate that the method requires a few minutes of computational time per image.

In a subsequent paper Catanzaro *et al*. [5] describe how the *gPb* method can be effectively parallelized and accelerated on a Graphics Processing Unit (GPU). Using their implementation one can perform *gPb* edge detection on a 0.15 Megapixel images in 1.8 seconds using a NVidia GTX 280 GPU. A speed up of over two orders of magnitude over a CPU based method.

This is an impressive result but it requires the use of a fairly powerful GPU subsystem and many commonly available computational platforms, such as laptops and cell phones, do not yet have access to that level of acceleration. The goal of this paper is to make accurate segmentation schemes based on normalized cuts more efficient by exploiting the structure of the underlying problem. The resulting ideas could be used to accelerate implementations on a variety of computational platforms including GPUs.

To this end this paper makes two distinct contributions firstly we characterize the performance of an edge detection scheme based on normalized correlation which was first proposed by Meer and Georgescu [11]. We propose a scheme for tuning the parameters of this method using training data and show that the resulting tuned detector produces results that are comparable to pB at a fraction of the computational cost. Secondly this paper propose a new variant of the normalized cuts segmentation scheme which solves a reduced order eigensystem that captures the essential features of the original problem. We show how to construct this reduced order system, explain its computational advantages and characterize its performance.

The remainder of this paper is organized as follows: Section 2 describes the proposed segmentation scheme in more detail while Section 3 presents experimental results obtained with an implementation of this method. Finally Section 4 discusses the conclusions that have been drawn so far.

2. Technical Approach

The overall approach to image segmentation embodied in this work is inspired by the work on *gPb* [3, 2] in that it starts with a set of edges derived from local pixel dif-

1063-6919/13 $26.00 © 2013 IEEE
DOI 10.1109/CVPR.2013.250

ferences and then invokes a globalization procedure which strengthens or weakens those edges based on an analysis of the eigenvectors produced by a normalized cuts procedure.

The procedure consists of three main processing stages: an edge extraction phase, a normalized cuts phase and a region merging phase which produces a segmentation result based on the local and global edge signals. The proposed modifications to the edge extraction and normalized cuts stages are discussed in the following subsections.

2.1. Normalized Edge Detection

The first step in the procedure is an edge extraction stage which produces a set of edgels. This system employs a variant of the method proposed by Meer and Georgescu [11] which can be thought of as computing the normalized cross correlation between each pixel window and a set of oriented edge templates.

Let $\mathbf{w}_{ij}^r \in \mathbb{R}^p$ denote the vector formed by stacking the intensity values within a circle of radius r pixels centered at pixel location, (i, j), and then subtracting the mean value. Let $\mathbf{t}_\theta^r \in \mathbb{R}^p$ denote the corresponding entries from an oriented step edge template. The entries in the oriented edge template are normalized so that the vector has unit length and zero mean, $\|\mathbf{t}_\theta^r\| = 1$, $\mathbf{1} \cdot \mathbf{t}_\theta^r = 0$. The Normalized Cross Correlation between the pixel window and the edge template can be computed as follows.

$$e_{ij}^r = \frac{|\mathbf{t}_\theta^r \cdot \mathbf{w}_{ij}^r|}{\|\mathbf{w}_{ij}^r\|} \tag{1}$$

This signal will range between 0 and 1 with higher values corresponding to windows which are well modeled by the step edge. The advantage of this approach is that it normalizes out the overall contrast associated with the window which helps to eliminate spurious responses in textured regions and amplify the response of low contrast step edges.

In this work we modify this edge detection scheme by introducing an additional factor in the denominator which is based on the average response to the template. Let μ_θ^r denote a scalar representing the average response to the edge template, $|\mathbf{t}_\theta^r \cdot \mathbf{w}_{ij}^r|$, over the entire image. The modified edge response s_{ij}^r is then given by the following expression.

$$s_{ij}^r = \frac{|\mathbf{t}_\theta^r \cdot \mathbf{w}_{ij}^r|}{(\|\mathbf{w}_{ij}^r\| + \beta_e \mu_\theta^r)} \tag{2}$$

The value of β_e used in the experiments was 5.

This modification serves to reintroduce some contrast information into the edge response so that larger steps have a greater response than smaller ones and slight variations in low contrast regions are not unduly amplified. The edge extraction procedure is applied to each channel of the Lab image and within each color channel we consider 4 different scales corresponding to radii of 2, 5, 10, and 20 pixels. This analysis step produces a total of twelve edge signal values at each pixel representing edge strengths in the different channels and scales. These values must then be combined into a single composite edge strength value for the pixel. Here we choose to compute a simple weighted sum of the edge signals where the weights are chosen through a regression procedure.

In the training phase we construct a labeled data set from the training images provided in the Berkeley Segmentation Database. We choose a subset of all of the pixels in the data set that were marked as edges to represent positive examples and choose an equal number of non-edge pixels to act as negative examples. An optimal set of weights was then determined by applying logistic regression to this data set.

The second row of Figure 5 shows the results of applying the edge extraction procedure to some sample images. Since this edge extraction procedure considers multiple scales it is able to give greater weight to more salient edges which appear at several scales while still responding to more subtle features that only appear at finer scales. These results are obtained by computing the maximum response over all orientations to produce an edge strength image and by applying non-maximal suppression to obtain thinned edges.

2.2. Reduced Order Normalized Cuts

In the original formulation of the Normalized Cuts segmentation procedure [14] the principal goal is to minimize the Rayleigh quotient given in Equation 3

$$\min_{\mathbf{y}} \frac{\mathbf{y}^T(D - W)\mathbf{y}}{\mathbf{y}^T D\mathbf{y}} \tag{3}$$

Where $\mathbf{y} \in \mathbb{R}^n$ is a vector with one element for each of the pixels in the image, $W \in \mathbb{R}^{n \times n}$ is a sparse, symmetric weight matrix whose entries reflect the affinities between the pixels and $D \in \mathbb{R}^{n \times n}$ is a diagonal matrix where the diagonal entries represent the sum of the weights of the edges impinging on each pixel.

Following [3] we construct the matrix W using the intervening contour cue. For each pixel, p, we consider every other pixel, q, within a radius of 5 pixels and find the largest edge signal value along the line connecting the two sites, s_{pq}. This value is then converted into an entry in the weight matrix via the following expression: $W_{pq} = \exp(-\rho s_{pq})$. The parameter ρ was set to 25 in our experiments.

The problem described in Equation 3 is ultimately solved, at least approximately, by finding the eigenvectors of the following generalized eigensystem corresponding to the k smallest eigenvalues where k is a constant on the order of 20.

$$(D - W)\mathbf{y} = \lambda D\mathbf{y} \tag{4}$$

Solving this generalized eigenvalue problem is typically the major computational bottleneck in the normalized cuts segmentation procedure. The prinicipal challenges are related to the size of the system and the fact that we are interested in recovering the smallest eigenvalues rather than the largest. We propose to construct a reduced order normalized cut system which will be easier to solve by replacing the vector $\mathbf{y}$ in the previous equations with the vector $L\mathbf{x}$ where $\mathbf{x} \in \mathbb{R}^m$, $L \in \mathbb{R}^{n \times m}$ and $k \ll m \ll n$. This leads to a new optimization problem:

$$\min_{\mathbf{x}} \frac{(L\mathbf{x})^T(D-W)(L\mathbf{x})}{(L\mathbf{x})^T D(L\mathbf{x})} \tag{5}$$

and a new generalized eigenvalue problem.

$$(L^T(D-W)L)\mathbf{x} = \gamma(L^T DL)\mathbf{x} \tag{6}$$

First we note that if $L\mathbf{x}$ is an eigenvector of the generalized eigensystem in Equation 4 then $\mathbf{x}$ will be an eigenvector of the system in Equation 6 with the same eigenvalue. From this we can conclude that if the k relevant eigenvectors of the original problem lie in the span of the matrix L then our reduced eigenector problem will actually produce exactly the same results as the original problem. That is the k smallest eigenvalues of the reduced problem will be the same as those of the original problem and the corresponding vectors $L\mathbf{x}_i$ will match the eigenvectors of the original problem modulo an irrelevant scale factor. More generally if the k eigenvectors are well approximated by the span of L then the eigenvectors of the reduced system will be good approximations for those of the original system.

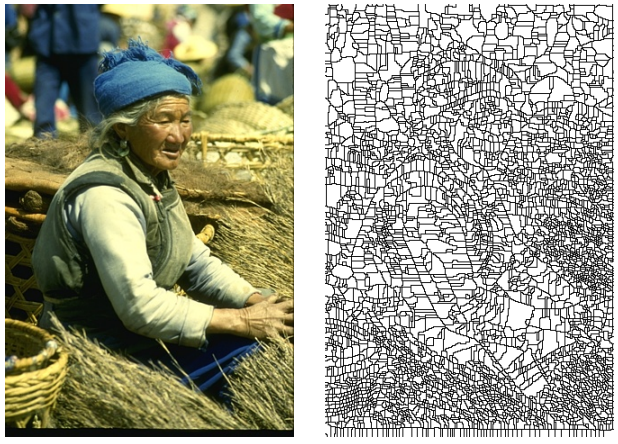

Figure 1. This figure shows the results of breaking an input image into superpixels by applying the watershed algorithm to the edge strength signal produced by the normalized edge detection stage.

Our aim then is to construct a matrix, L, whose column span captures the variation we expect in the eigenvectors of the orginal system. We do this by first considering the edge strength signals produced in the edge detection phase. We produce an edge strength image by simply recording the max edge response over all orientations at each pixel as described earlier. We then apply a watershed transform to this edge strength image to produce a set of superpixels as shown in Figure 1. In our experiments this step typically broke the image into around 5000 segments.

The purpose of the L matrix is to assign a value to each pixel based on the labels assigned in its neighborhood. The simplest approach would be to assign each pixel the value associated with its label, this would correspond to a binary L matrix with a single 1 entry in each row. The approach used in this paper is more nuanced, each pixel constructs a weighted average of the values in the label vector, $\mathbf{x}$, by looking at its label and the labels assigned to its neighbors. In the experiments we considered all of the neighbors that fell within a radius of 3 pixels. Each of the neighbors is assigned a weight based on its position, this weight falls of exponentially with distance. Pixels that lie on watershed boundaries are assigned zero weights. Weights associated with neighbors that share the same label are summed. Finally the label weights associated with each pixel are normalized so that they sum to 1. The end result is an L matrix whose span produces images that are smooth in textureless areas but can vary sharply near potential discontinuities. We have found in practise that the L matrices constructed in this manner do a significantly better job of capturing the desired eigenvectors than the binary versions.

Once the L matrix has been constructed we can turn our attention to solving the reduced order eigensystem described in Equation 6. We first note that this system is much smaller than the original system since m is on the order of 5000 where n was on the order of 115000 for the images in our test set.

The second reason that this system is more amenable to analysis is more subtle. First we observe that the reduced order eigensystem is still sparse, if we consider the matrices $W' = L^T W L$ and $D' = L^T D L$ we note that these matrices will essentially capture the adjacency structure of the watershed superpixels in the image.

Next observe that we are ultimately interested in finding the eigenvectors associated with the *smallest* eigenvalues of the generalized eigensystem. These eigenvectors can be found using a variant of the Lanczos procedure [10] which is related to the power method and proceeds by repeatedly performing matrix vector multiplications to produce a set of vectors that capture the desired portion of the spectrum. Here we note that if we are interested in finding the *smallest* eigenvalues of a symmetric positive semi definite matrix A we can improve our convergence rate substantially if we consider instead A^{-1} since the smallest eigenvalues of A will correspond to the largest eigenvalues of A^{-1} and applying the power method to A^{-1} will quickly produce the

eigenvectors and eigenvalues that we are most interested in.

Finally we note that the matrices $(W' - D')$ and D' are quite amenable to factorization using a modified Cholesky decomposition. More specifically we have found that if we use the Approximate Minimum Degree algorithm [1] to reorder the rows and columns of these matrices they can be factored quite readily into sparse Cholesky factors. This is important because it means that we can apply the inverses of $(W' - D')$ and D' very rapidly which is what is required to apply the accelerated Lanczos method described above. Note that we do not need to compute the actual inverses of $(W' - D')$ or D', we only need to be able to apply these inverses to vectors, a task for which a sparse Cholesky decomposition is ideally suited.

As an example consider the image shown in Figure 1. The watershed algorithm produced a superpixel segmentation containing 4329 segments. The associated matrix W' had a total of 131943 entries. Our modified Cholesky factorization produced a lower triangular matrix that had a total of 290812 entries. Since the computational complexity of our Lanczos procedure will be governed by the fill structure of the Cholesky factors this is a quite advantageous result.

We note that the approach proposed here differs from the scheme described by Fowlkes et al. [9]. The goal in that work was to develop a scheme that could be used to approximate the eigenvectors of a weight matrix that considered the association between every pair of pixels in the image. Computing the eigenvectors of such a large system was decidedly unappealing so the authors propose an approach wherein they randomly sampled a subset of the rows of the affinity matrix to form a smaller, dense system involving only a few hundred rows. They then analyzed the spectral properties of that smaller system and used them to approximate the full system.

The approach advocated in this paper leverages the observation that in this image segmentation task the edge signal provides useful information about the final structure of the eigenproblem. By constructing a basis tailored to the content of the image we are able to identify a subspace that captures the nuances of the edges and the details found in the full system.

Once the eigenvectors have been computed, the segmentation scheme proceeds along the lines described in [3]. Each eigenvector is rescaled so that the maximum and minimum values correspond to 1 and 0. The vectors are then scaled by the square root of the associated eigenvalue to reflect their relative importance. Finally we compute the absolute values of the directional derivatives of each of the eigenimages and sum them to produce a spectral edge strength signal, $s_g(x, y, \theta)$. This spectral signal is combined with the oriented edge strength signal produced in the edge extraction phase, $s_{ncc}(x, y, \theta)$, to produce a final edge strength signal. $s_f(x, y, \theta) = \sigma_l s_{ncc}(x, y, \theta) + s_g(x, y, \theta)$. In our experiments we chose to recover the first 31 eigenvectors and σ_l was set to 400.

The resulting final oriented edge strength signal, $s_f(x, y, \theta)$, was passed to the **contours2ucm** routine provided as part of the Berkeley Segmentation Database to compute the final ultrametic contour map.

3. Experimental Results

The proposed edge detection and segmentation schemes were evaluated using the Berkeley Segmentation Database, BSDS, with the methods described in [3]. The parameters of the segmentation procedure were tuned using the 200 images in the training set and the final procedure was then applied to both the test and validation sets.

Figure 2 plots the precision recall curve for the Normalized Edge Detection scheme proposed in Section 2.1 on the BSDS300 data set and compares its response to that of the Pb and mPb detectors described in [3]. Note that the proposed scheme produces results which are comparable to those of the Pb detector on aggregate on this dataset but at a much lower computational cost. The mPb detector provides somewhat better performance with an F score of 0.67.

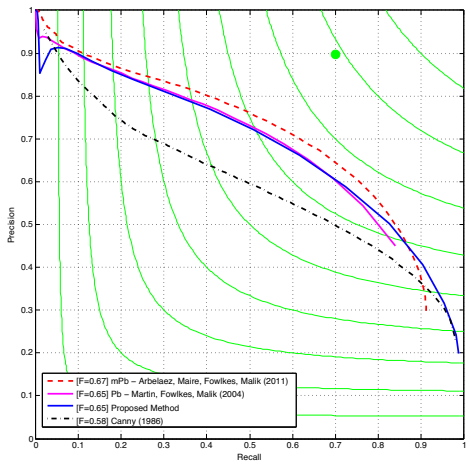

Figure 2. The precision recall curve of the proposed normalized edge detection scheme on the boundary benchmark of the BSDS 300 evaluation test set.

Figures 3 and 4 compare the performance of the proposed reduced order segmentation scheme to other segmentation schemes in the literature on the BSDS 300 and BSDS 500 data sets. Table 1 compares the method to other state of the art methods based on the precision recall performance in identifying salient boundaries. Table 2 summarizes the performance of the method using a variety of criterion including the Covering of ground truth segments, the Probablistic Rand Index (PRI) and the Variation of Information (VI) measures.

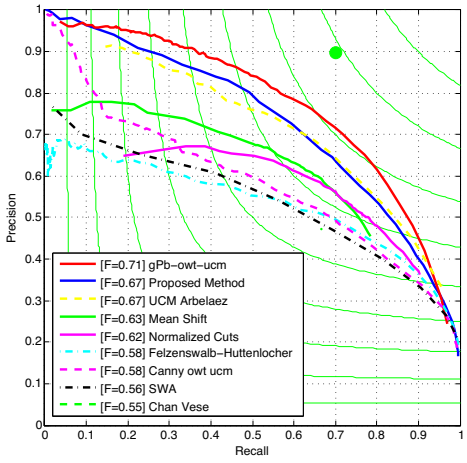

Figure 3. This graph compares the performance of the proposed reduced order normalized cuts segmentation scheme to other methods on the boundary benchmark of the BSDS 300 evaluation test set.

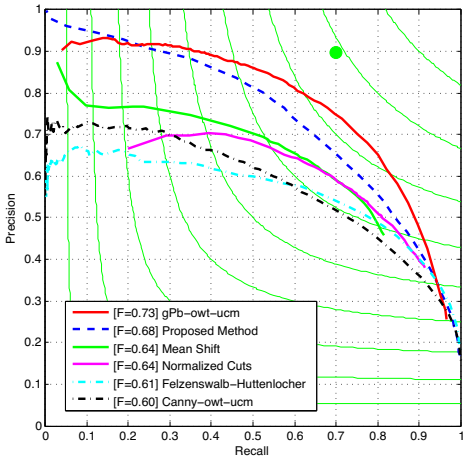

Figure 4. This graph compares the performance of the proposed reduced order normalized cuts segmentation scheme to other methods on the boundary benchmark of the BSDS 500 evaluation test set.

Taken together, these results indicate that the performance of the proposed method is significantly better than every other method except the gPb-owt-ucm algorithm and the recently proposed method of Ren and Bo [12].

Figure 5 shows the results of running the proposed scheme on a few of the images in the data set. The first row shows the input image, the second row shows the results of the normalized edge detection stage. The third row shows the ultrametric contour maps produced by the **contours2ucm** stage. The segmentation results at the optimal data scale (ODS) and optimal image scale (OIS) are shown in the final two rows.

	BSDS 300			BSDS 500		
	ODS	OIS	AP	ODS	OIS	AP
Human	0.79	0.74	-	0.80	0.80	-
Ren and Bo [12]				0.74	0.76	0.77
gPb-owt-ucm [3]	0.71	0.74	0.73	0.73	0.76	0.73
Proposed Method	0.67	0.72	0.73	0.68	0.72	0.74
Mean Shift [6]	0.63	0.66	0.54	0.64	0.68	0.56
NCuts [7]	0.62	0.66	0.43	0.64	0.68	0.45
Canny-owt-ucm [3]	0.58	0.62	0.53	0.60	0.64	0.58
Felz-Hutt [8]	0.58	0.62	0.53	0.61	0.64	0.56
SWA [13]	0.56	0.59	0.54	-	-	-
gPb	0.70	0.72	0.66	0.71	0.74	0.65
Canny	0.58	0.62	0.58	0.60	0.63	0.58

Table 1. Results obtained on the BSDS data set on the boundary benchmarks using the evaluation methodology suggested in [3, 2]. The values in the table correspond to the optimal harmonic means obtained over the precision recall. ODS refers to the optimal data scale and OIS to the optimal image scale. AP refers to the average precision. The values are reproduced from the tables in [3].

	BSDS 300						
	Covering			PRI		VI	
	ODS	OIS	Best	ODS	OIS	ODS	OIS
Human	0.73	0.73	-	0.87	0.87	1.16	1.16
gPb-owt-ucm [3]	0.59	0.65	0.75	0.81	0.85	1.65	1.47
Proposed Method	0.56	0.62	0.72	0.79	0.84	1.74	1.53
Mean Shift [6]	0.54	0.58	0.66	0.78	0.80	1.83	1.63
Felz-Hutt [8]	0.51	0.58	0.68	0.77	0.82	2.15	1.79
Canny-owt-ucm [3]	0.48	0.56	0.66	0.77	0.82	2.11	1.81
SWA [13]	0.47	0.55	0.66	0.75	0.80	2.06	1.75
NCuts [7]	0.44	0.53	0.66	0.75	0.79	2.18	1.84
Chan Vese [4]	0.49	-	-	0.75	-	2.54	-
Quad-Tree	0.33	0.39	0.47	0.71	0.75	2.34	2.22

	BSDS 500						
	Covering			PRI		VI	
	ODS	OIS	Best	ODS	OIS	ODS	OIS
Human	0.72	0.72	-	0.88	0.88	1.17	1.17
gPb-owt-ucm [3]	0.59	0.65	0.74	0.83	0.86	1.69	1.48
Proposed Method	0.56	0.62	0.73	0.81	0.85	1.78	1.56
Mean Shift [6]	0.54	0.58	0.66	0.79	0.81	1.85	1.64
Felz-Hutt [8]	0.52	0.57	0.69	0.80	0.82	2.21	1.87
Canny-owt-ucm [3]	0.49	0.55	0.66	0.79	0.83	2.19	1.89
NCuts [7]	0.45	0.53	0.67	0.78	0.80	2.23	1.89

Table 2. Results obtained on the BSDS data set on the region benchmark using the evaluation methodology suggested in [3, 2]. The first 3 columns report the score on the covering of the ground truth segments at the optimal data scale (ODS), and the optimal image scale (OIS). The next two columns report the score according to the Probabilistic Rand Index (PRI). The final 2 columns report the score with respect to the Variation of Information criterion. The values are reproduced from the tables in [3].

Table 3 indicates the time taken by the four main phases of the proposed segmentation scheme in seconds. The scheme was implemented in Matlab with a combination of Matlab functions and mex files. Importantly, the entire computation can be carried out in under 10 seconds on a standard laptop in Matlab without any GPU acceleration. For comparison Table 4 summarizes the time required to perform the major steps of the gPb-owt-ucm algorithm on the same computer. Here the time taken to perform a segmentation is on the order of 4 minutes. The timings were performed on a Macbook Pro with a 2.2 GHz quad-core In-

Figure 5. Segmentation results produced by the proposed method. The first row contains the input image, the second the results of the Normalized Edge Detector, the third the ultrametric contour maps which include the globalized edge weights. The fourth and fifth rows show segmentations at the ODS and OIS respectively.

tel Core i7 processor.

Note that the reduced order normalized cut scheme allows us to compute the 31 smallest eigenvectors within 0.42 seconds in Matlab, this is markedly better than the 0.777 seconds reported to recover 17 eigenvectors with an optimized GPU implementation [5].

In order to investigate how the reduced order approximation affected the quality of the final results. We ran an experiment on a subset of the dataset without using this optimization. That is we computed the eigenvectors of the original unreduced system and used those results for the final segmentation. We then evaluated the quality of the resulting image segmentations by computing the precision recall curve for the boundary detection task. The scores, curves and results were identical to those obtained with the reduced order system. The only difference was time, finding the eigenvectors of the full system required on average 430 seconds while solving the reduced system required only 0.4 seconds. This suggests that the reduced order system effectively captures the essential structure of the full eigenproblem for this task.

Further optimizations are certainly possible. The edge detection scheme was implemented in a straightforward manner using Matlab's convolution routines. A better implementation could make use of integral images which would allow us to avoid convolutions entirely and would drastically simplify the issue of computing responses over multiple scales. A real time implementation of the normalized edge detection scheme is a distinct possibility.

Similarly a more careful C implementation could avoid some of the inefficiencies associated with Matlab's sparse matrix operations and would improve the running times of

Phase	mean	median	max
Normalized Edge Detection	2.0309	2.0245	2.2511
Computing Weight Matrix	2.5776	2.5738	2.7515
Reduced Order Normalized Cuts	0.4171	0.4039	1.6866
contour2ucm	6.5531	6.5668	11.0380

Table 3. This table shows the mean, median and maximum time spent in seconds on each of the four major computational phases when the proposed segmentation procedure was run on the 200 images in the BSDS 500 test set.

Phase	mean time in seconds
mPb - computing local cues	61.13
mPb - smoothing cues	8.50
Spectral Pb	162.75

Table 4. This table shows the average time spent in seconds on each of the major computational phases of the gPb segmentation algorithm

the weight matrix construction phase considerably.

4. Conclusions and Future Work

The goal of this paper has been to explore approaches to accelerating edge detection and segmentation schemes based on the gPb framework by exploiting the underlying structure of the problem. To this end we have investigated the performance of an edge extraction scheme based on normalized correlation and shown that, with appropriate tuning, the scheme can be made to produce competitive results without excessive computation.

This paper also presents ideas for tackling the key bottleneck in the normalized cuts procedure, the computation of generalized eigenvectors. We propose a reduced order normalized cut problem which is designed to capture the essential features of the full scale problem. The key to this approach is the construction of an appropriate matrix L whose columns provide a good approximation for the eigenvectors of the original system. We propose a scheme for constructing such a matrix using an initial superpixel segmentation of the frame and note that the resulting reduced systems exhibit special structure which can be exploited in the eigenvector analysis procedure to further accelerate convergence. Timing results show that this approach can provide a dramatic improvement in performance.

Note that the ideas presented in this paper could be employed independently. The normalized edge detection scheme could be used for purposes other than segmentation and the reduced order acceleration could be applied to the output of other edge detectors. As an example it would be entirely possible to use the output of the mPb detector as an input to a reduced order normalized cuts system. Similarly one can imagine using other superpixel segmentation schemes to construct appropriate L matrices that may provide other benefits. We plan to explore such combinations in future work.

Acknowledgments

This research was partially sponsored by the Army Research Laboratory Cooperative Agreement Number W911NF-10-2-0016.

References

[1] P. R. Amestoy, T. A. Davis, and I. S. Duff. Algorithm 837: Amd, an approximate minimum degree ordering algorithm. *ACM Trans. Math. Softw.*, 30(3):381–388, Sept. 2004.

[2] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. From contours to regions: An empirical evaluation. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2294 –2301, 2009.

[3] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(5):898–916, May 2011.

[4] L. Bertelli, B. Sumengen, B. S. Manjunath, and F. Gibou. A variational framework for multiregion pairwise-similarity-based image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(8):1400–1414, August 2008.

[5] B. Catanzaro, B.-Y. Su, N. Sundaram, Y. Lee, M. Murphy, and K. Keutzer. Efficient, high-quality image contour detection. In *IEEE International Conference on Computer Vision*, pages 2381–2388, October 2009.

[6] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:603–619, 2002.

[7] T. Cour, F. Benezit, and J. Shi. Spectral segmentation with multiscale graph decomposition. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 1124 – 1131 vol. 2, 2005.

[8] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *Int. J. Comput. Vision*, 59(2):167–181, 2004.

[9] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the nystrom method. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(2):214–225, 2004.

[10] G. H. Golub and C. F. V. Loan. *Matrix Computations*. The Johns Hopkins University Press, 3rd edition, 1996.

[11] P. Meer and B. Georgescu. Edge detection with embedded confidence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):1351–1365, December 2001.

[12] X. Ren and L. Bo. Discriminatively trained sparse code gradients for contour detection. In *Neural Information Processing Systems (NIPS)*, December 2012.

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

[14] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 1997.