

Evaluating image segmentation algorithms using monotonic hulls in fitness/cost space

Mark Everingham, Henk Muller and Barry Thomas *
Advanced Computing Research Centre
University of Bristol
Bristol, BS8 1UB, UK.
everingm@cs.bris.ac.uk

Abstract

Image segmentation is the first stage of processing in many practical computer vision systems. While development of particular segmentation algorithms has attracted considerable research interest, relatively little work has been published on the subject of their evaluation. In this paper we propose a framework for quantitative evaluation of segmentation algorithms which we believe addresses shortcomings of previous approaches, and use this framework to compare several state-of-the-art algorithms.

1 Introduction

Image segmentation is the first stage of processing in many practical computer vision systems. Applications include image interpretation [9, 4] and searching image databases by content [2]. Over the last few decades many segmentation algorithms have been developed, with the number growing steadily every year. In contrast, relatively little effort has been spent in attempting to evaluate the effectiveness of such algorithms. Typically the effectiveness of a new algorithm is demonstrated only by the presentation of a few segmented images. This allows only subjective and qualitative conclusions to be drawn about that algorithm. We believe that if segmentation algorithms are to be successfully applied in real vision systems, quantitative assessment methods of algorithms need to be defined.

2 Previous evaluation methods

The main difficulty in assessing image segmentation algorithms stems from the ill-defined nature of the problem being addressed. In his survey on evaluation methods for image segmentation [14], Zhang proposes this definition of image segmentation:

“[Image segmentation] consists of subdividing an image into its constituent parts and extracting these parts of interest (objects).” [14]

*Part of this work was sponsored by Hewlett-Packard Research Laboratories, Bristol

This captures the procedural quality of segmentation but leaves many unanswered questions, notably how to define “constituent parts” and “parts of interest”, and how to assess the degree of success or failure of an algorithm in the case that it does not perform optimally. Inevitably these questions will have to be answered in an application-dependent manner. They relate to what subsequent stages of processing are to be applied to the results of segmentation in order to achieve the goal of the entire vision system. However, it is rarely feasible to build entire systems in order to test different segmentation algorithms because of expense, and because the properties of the segmentation algorithm will often determine what form subsequent processing should take. Therefore we are interested in evaluating segmentation algorithms without implementation of subsequent processing.

Zhang [14] proposes a classification of evaluation methods as “analytical”, “empirical goodness”, or “empirical discrepancy”.

“Analytical” methods attempt to characterize an algorithm itself in terms of principles, requirements, complexity etc. without reference to a concrete implementation of the algorithm or test data. For example one can define the time complexity of an algorithm or its response to a theoretical data model such as a step edge. While in domains such as edge detection this may be useful, in general the lack of a general theory of image segmentation limits these methods.

“Empirical goodness” methods evaluate algorithms by computing a “goodness” metric on the segmented image *without a priori* knowledge of the desired segmentation result. For example Levine and Nazif [10] use a measure of intra-region grey-level uniformity as their goodness metric, assuming that in a well-segmented image regions should have low variance of grey-level. The advantage of this class of methods is that they require only that the user defines a goodness metric, do not require manually segmented images to be supplied as ground truth data, and can be used in an online manner so that the effectiveness of an algorithm can be monitored during actual application. The great disadvantage is that the goodness metrics are at best heuristics, and may exhibit strong bias towards a particular algorithm. For example the intra-region grey-level uniformity metric will cause any segmentation algorithm which forms regions of uniform texture to be evaluated poorly.

“Empirical discrepancy” methods calculate a measure of discrepancy between the segmented image output by an algorithm and the correct segmentation desired for the corresponding input image. In the case of synthetic images the correct segmentation can be obtained automatically from the image generation procedure while in the case of real images it must be produced manually or semi-automatically by an experienced operator. This is the main disadvantage in that manual creation of ground truth data is time-consuming. Analogous to the case of the “empirical goodness” methods, a discrepancy measure must be explicitly defined, but this is likely to be easier to do and exhibits less bias than the former methods because of the availability of ground truth data.

Several discrepancy measures have been proposed. One of the earliest and most intuitive measures [13] treats the segmentation task as a multi-class classification problem where each pixel has an associated correct class and takes measures of classification error from the pixel-wise class confusion matrix. Other discrepancy measures calculate the distances between mis-segmented pixels and the nearest correctly segmented pixels [13] thus introducing a spatial component to the measure, or are based on differences between feature values measured from regions of the correctly segmented and output images [15].

2.1 Limitations

We believe the main problem with previous approaches is their attempt to capture an algorithm's effectiveness in a single metric. There are two aspects to this problem. First, a published algorithm will almost always have several parameters which need to be set by the user so that an algorithm does not define a single segmentation result for a given input image, but rather a set of results which can be chosen from by selection of the algorithm parameters. Second, in reality we expect that we always have to make a trade-off between different properties of a segmentation algorithm, for example the quality of the segmentation versus the execution time. Measures which are based solely on a quality metric do not allow such trade-offs to be evaluated objectively.

Our approach, described in the next section, does not fit easily into any one of Zhang's categories [14] but can be seen as a unification of all categories into a consistent framework, defining a general methodology of comparison incorporating multiple measures rather than advocating the use of a single particular measure. Our approach is most similar to the "empirical discrepancy" methods but importantly has the distinction of not defining just a single discrepancy metric and evaluating effectiveness in a discrepancy/parameter space but instead performing evaluation in a multi-dimensional fitness/cost space.

3 Evaluation in fitness/cost space

Rather than attempt to define very specific measures of a segmentation algorithm's properties we start with a very general form of overall fitness function

$$H(a_{\vec{p}}, I) = \Phi(f_1(a_{\vec{p}}, I), \dots, f_m(a_{\vec{p}}, I), c_1(a_{\vec{p}}, I), \dots, c_n(a_{\vec{p}}, I)) \quad (1)$$

where $a_{\vec{p}}$ is a segmentation algorithm a with parameters \vec{p} and I is a set of ground truth images. Functions $f_i(a_{\vec{p}}, I)$ are individual fitness functions defined to increase monotonically with the fitness of some particular aspect of the algorithm's behavior. Functions $c_i(a_{\vec{p}}, I)$ are individual cost functions defined to increase monotonically with the cost of some particular aspect of the algorithm's behavior. Cost functions c_i could equivalently be defined as negative fitness functions but we make the distinction here for the sake of clarity.

Φ combines the individual fitness and cost functions into an overall measure of fitness. For example if $f_1(a_{\vec{p}}, I)$ measures the accuracy of the segmentation and $c_1(a_{\vec{p}}, I)$ the running time of the algorithm, an appropriate form for Φ might be some nonlinear weighted sum of the two. We believe that in general it is difficult to specify an exact form for Φ since this requires defining exact trade-offs to be made between fitness and costs, and therefore assume it to be of unknown form. With no loss of generality we assume that Φ increases monotonically with increasing values of all fitness functions f_1, \dots, f_m and decreases monotonically with increasing values of all cost functions c_1, \dots, c_n .

With Φ thus unconstrained, the behavior of an algorithm a applied to a set of images I with a particular choice of parameters \vec{p} can be characterized by a point in the $m + n$ -dimensional "fitness/cost" space defined by evaluation of the fitness and cost functions. Varying the parameters \vec{p} of the algorithm a produces a new point in the space as the fitness and cost parts of the overall fitness change. By this representation we decouple the fitness/cost trade-off from the particular parameter set used by an individual algorithm.

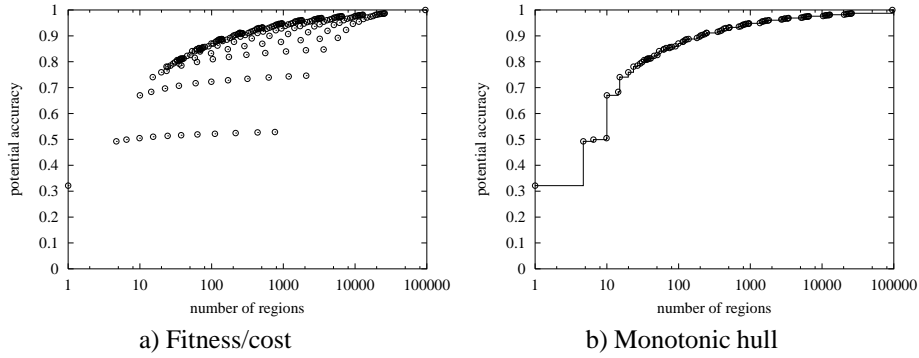


Figure 1: Example fitness/cost graph and monotonic hull

We can plot a projection of Φ onto a single fitness function f_i and cost function c_j for different parameters \vec{p} by a zero-scaling of all other functions. Figure 1a shows an example where a potential accuracy function f is plotted against the mean number of regions c output per image for a particular segmentation algorithm. The points represent different parameters \vec{p} , in this case there being two parameters to the segmentation algorithm. In this example we use the number of regions as a cost function in that it characterizes the processing time required for subsequent processing of the segmented image assuming a constant processing time per region. We shall consider suitable fitness and cost functions in more detail in Section 4.1.

3.1 Monotonic hull

Many of the points on the graph of Figure 1a are intuitively undesirable since they have lower fitness and higher cost than points corresponding to other choices of parameters. In the 2-D case these undesirable parameter settings are points which are to the bottom-right of another point. Since we have defined our overall fitness function Φ to be monotonic, then for a particular algorithm a with parameters taken from the set P_a it is readily shown that the only worthwhile choices of parameters are in the set

$$\{\vec{p} \in P_a \mid \neg \exists \vec{q} \in P_a : n(a_{\vec{q}}, a_{\vec{p}})\} \quad (2)$$

where

$$n(a_{\vec{q}}, a_{\vec{p}}) = \begin{cases} \text{true} & \text{if } \forall f \in F : f(a_{\vec{q}}, I) \geq f(a_{\vec{p}}, I) \wedge \forall c \in C : c(a_{\vec{q}}, I) \leq c(a_{\vec{p}}, I) \\ \text{false} & \text{otherwise} \end{cases} \quad (3)$$

and $F = \{f_1 \dots f_m\}$, $C = \{c_1 \dots c_m\}$. We term such points the “monotonic hull”. Figure 1b shows these points for the graph of Figure 1a, where we have drawn connecting lines to indicate the partitioning of the space by the hull.

The strength of this construction is that we can readily show that for *any* choice of overall fitness function Φ that increases monotonically in F and decreases monotonically in C , parameters which do not correspond to points on the hull are *always* bad choices since there is another point on the hull which increases the overall fitness.

We can extend Equation 2 to the case of multiple algorithms and define the monotonic hull over both algorithms a and their parameters \vec{p} :

$$\{a_{\vec{p}} | a \in A \wedge \vec{p} \in P_a \wedge \neg \exists b \in A, \vec{q} \in P_b : n(b_{\vec{q}}, a_{\vec{p}})\} \quad (4)$$

This has the natural consequence that any algorithm which does not fall on the monotonic hull is similarly a bad choice for *any* monotonic fitness function Φ . An example will be shown in Figure 3a.

3.2 Convex hull

Thus far the only limitation we have imposed on the form of the overall fitness function Φ is that it be monotonic. If we know that Φ is a *linear* function of the fitness and cost functions

$$\Phi(f_1(a_{\vec{p}}, I), \dots, f_m(a_{\vec{p}}, I), c_1(a_{\vec{p}}, I), \dots, c_n(a_{\vec{p}}, I)) = \sum_{i=1}^m \alpha_i f_i(a_{\vec{p}}, I) - \sum_{j=1}^n \beta_j c_j(a_{\vec{p}}, I) + k \quad (5)$$

where without loss of generality $\forall i : \alpha_i \geq 0$ and $\forall j : \beta_j \geq 0$, then we can readily show that all worthwhile algorithms and parameters fall on the $m+n$ -dimensional *convex* hull [1]. Known parameters α and β of the convex fitness function define an iso-fitness plane through the $m+n$ -dimensional fitness/cost space such that all points on that plane have equal overall fitness. For example in the 2-D case of Figure 1 a fitness function $\Phi = \alpha f(a_{\vec{p}}, I)$ would define a horizontal iso-fitness line such that all algorithms resulting in equal potential accuracy have equal overall fitness regardless of the number of regions.

The 2-D convex hull has been used in the analysis of receiver operating characteristic (ROC) curves for binary classifiers [12]. In this case the true positive versus false positive rate of the classifier defines a point in a 2-D fitness/cost space, and a linear weighting of the two is easily justifiable. In the more general case we believe it is often hard to justify the use of a linear weighting.

4 Experimental results

Having defined the N -dimensional monotonic hull we have compared seven segmentation algorithms using four metrics and three sets of ground truth images [6]. Due to space restrictions here we present the results of three metrics and one set of ground truth images (the full results support our conclusions).

4.1 Fitness and cost functions

In our experiments we define the objective of segmentation to be to form regions such that if the optimal object label is assigned to all pixels of a region, as much as possible of the object set is correctly labelled. The object label is a unique identifier of each object present in the images. The test data is a set of color images of outdoor urban scenes and their manual segmentation into objects according to a set of eight classes including road, pavement, car etc. Our fitness function expressing *potential* accuracy for a single image

is the following:

$$f(a_{\vec{p}}, i) = \frac{\sum_{o \in O_i} |\{x \in o | \hat{o}_x = o\}| w(o)}{\sum_{o \in O_i} w(o)} \quad (6)$$

where O_i is the set of objects in image i , an object $o \in O_i$ is a set of pixels, and \hat{o}_x is the optimal object label of a pixel x which maximizes the overall measure. The contribution of each object to the measure is weighted by a function w for which we have tried several forms. Setting $w(o) = 1$ gives the proportion of image pixels that could be correctly labelled. This is the same as has been used in other methods [13] but may introduce bias in that objects or classes of object having many pixels contribute more to the measure than those with fewer pixels. We can remove this bias by defining w thus:

$$w(o) = \begin{cases} \frac{1}{P(\lambda_o)|o|} & \text{if } |o| \geq k \\ 0 & \text{if } |o| < k \end{cases} \quad (7)$$

where λ_o is the class label of an object and $P(\lambda)$ the prior probability of an object being drawn from class λ . The constant k causes objects with smaller pixel area than k to be discarded which prevents very small objects biasing the metric. We must define k manually but it can remain constant across different images and segmentation algorithms. In this form we measure the *proportion* of each object which can potentially be correctly labelled. We term un-normalized fitness f_u and normalized by Equation 7 f_n .

We define three cost functions: c_r =number of regions, c_t =segmentation time, and a third c_l which represents the information about an object which is not present in a single region:

$$c_l(a_{\vec{p}}, i) = 1 - \frac{\sum_{r \in R_i} |\{x \in r | \hat{o}_r = o_x\}| w(\hat{o}_r)}{\sum_{r \in R_i} w(\hat{o}_r)} \quad (8)$$

where R_i is the set of segmented regions in image i , \hat{o}_r is the optimal object label for a region (defined to maximize Equation 6), and o_x is the correct object label for pixel x . With $w(r, \hat{o}_r) = 1$ this measures the number of pixels of an object which are *not* in a given region to be optimally labelled as that object, biased as described in Section 4.1. We can normalize using Equation 7 again and obtain the *proportion* of pixels of an object which are not part of a given region. We term this un-normalized cost c_{ul} and normalized by Equation 7 c_{nl} . In either form we have a measure of the information about an object which is not available in a particular single region for subsequent stages of processing, assuming that information about an object is uniformly distributed across its pixels.

4.2 Segmentation algorithms

We ran experiments with seven segmentation algorithms chosen to represent a cross-section of state-of-the-art algorithms for which implementations are publicly available, and simple or fast algorithms. BLK simply divides an image statically into square blocks of constant size using no image information. It has the advantage of zero run-time since the segmentation is static, independent of the input image, and the algorithm is useful as a baseline. KMG, KMC, and GMT use a common approach of clustering on low-level features, forming connected regions and merging regions until a minimum region size

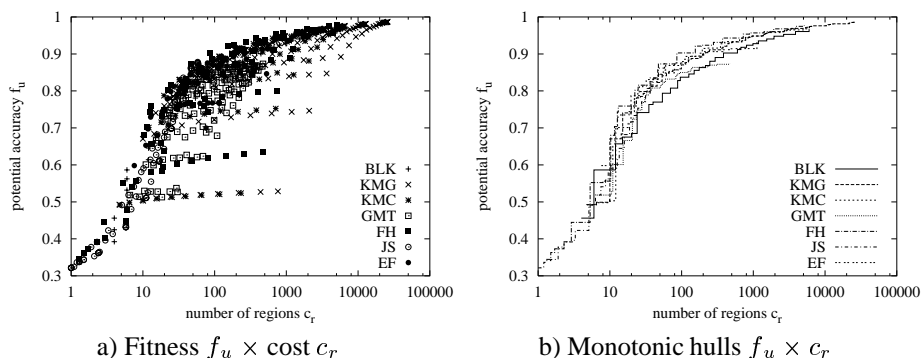


Figure 2: Fitness $f_u \times \text{cost } c_r$ and monotonic hulls

is obtained. KMG uses grey-level features alone and the K-means clustering algorithm [3]. KMC uses color features and K-means. GMT uses color and texture features from a bank of 8 Gabor filters [8] and a Gaussian mixture model [3] for clustering. FH [7] uses dynamic programming to form regions which are guaranteed to be neither too coarse nor fine with respect to a color edge strength measure within and between regions, then merges regions to a minimum region size. JS [5] uses color quantization followed by a multi-scale region growing step which aims to segment both uniform and textured regions. EF [11] uses predictive coding to estimate the direction of change in color and Gabor texture features, and forms boundaries by propagating the flow field.

4.3 Results

Figure 2a shows all algorithms and parameters tried in the f_u/c_r space, and Figure 2b the monotonic hulls for each algorithm. Fitness and cost were taken as the mean across all images in the test set. We restrict ourselves to use of the monotonic rather than convex hull since we cannot be sure that the linearity assumptions of the latter are justified here. Figure 3a shows the monotonic hull over all algorithms. Here we have restricted the y-axis to minimum 50% potential accuracy and the number of regions to maximum 10,000 for the sake of clarity since we argue that a segmentation scheme outside these ranges is unlikely to be useful.

Algorithms which have *no* points on the hull are shown grayed out in the key. Interestingly, segmenting the entire image into one region gives a potential accuracy of over 30%, indicating the bias in metric f_u . 80% of the image can potentially be labelled correctly using 25 regions, and 155 regions are required to reach 90%. We can see that the hull is dominated by FH. The two algorithms which use texture features (GMT and EF) both perform badly, with GMT contributing no points to the hull and EF only two, suggesting that texture is not useful in this domain. (In other experiments using texture mondrians we found EF also performed badly compared to other algorithms [6]). JS also performs unconvincingly. We found a general problem with both the EF and JS algorithms to be difficulty in obtaining sufficient numbers of output regions.

Figure 3b shows the monotonic hull for the normalized f_n fitness measure. Two things are noteworthy: many more regions are required to achieve potential accuracy according

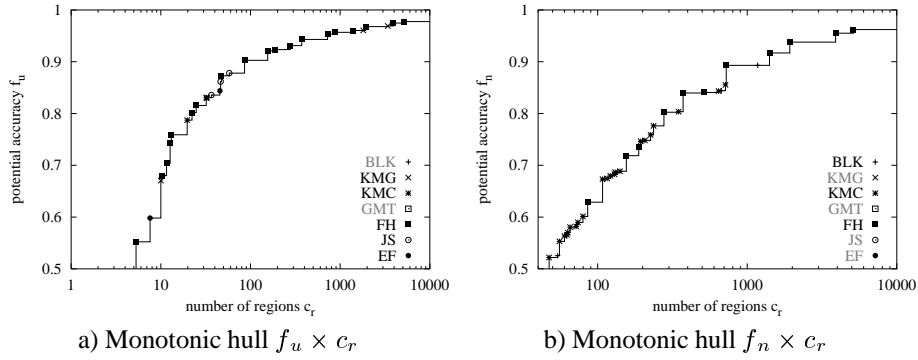


Figure 3: Monotonic hulls fitness $f_u \times \text{cost } c_r$ and fitness $f_n \times \text{cost } c_r$

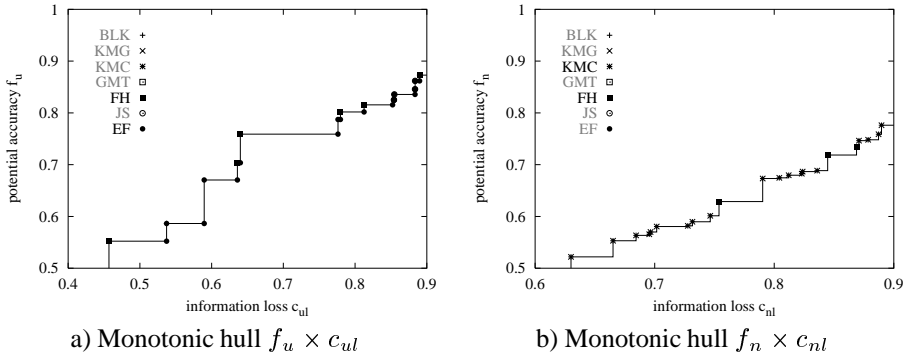


Figure 4: Monotonic hulls fitness $f_u \times \text{cost } c_{ul}$ and fitness $f_n \times \text{cost } c_{nl}$

to the normalized metric, suggesting some bias in the algorithms towards larger regions, and now only three algorithms lie on the hull: FH, KMC and BLK. BLK's presence on the hull is interesting, suggesting that for very low or high numbers of regions its performance may be comparable to more principled algorithms. In addition we see that the simple color K-means algorithm KMC now seems equal to FH.

Figure 4 shows the monotonic hull for un-normalized and normalized information loss metrics (Equation 8). For the un-normalized (pixel area) measure only FH and EF are on the hull, while for the normalized (proportion of object) measure FH appears but KMC dominates. These results suggest that for large objects both FH and EF are able to produce accurate object boundaries while not grossly over-segmenting objects, but for small objects KMC performs better. Note however that for both measures the information loss for a reasonable potential accuracy is high, for example to potentially achieve 80% accuracy by pixels each region contains on average around only 20% of an object. This suggests that classification of objects using single regions may be unrealistic.

Thus far we have only presented results using 2-D spaces for ease of presentation. We conclude with an evaluation of the algorithms in the 3-D $f_u \times c_r \times c_t$ space. This allows evaluation of the algorithms' potential accuracy, output complexity, and processing time. Figure 5 shows a view of the resulting monotonic hull. Only KMC, BLK, FH and KMG

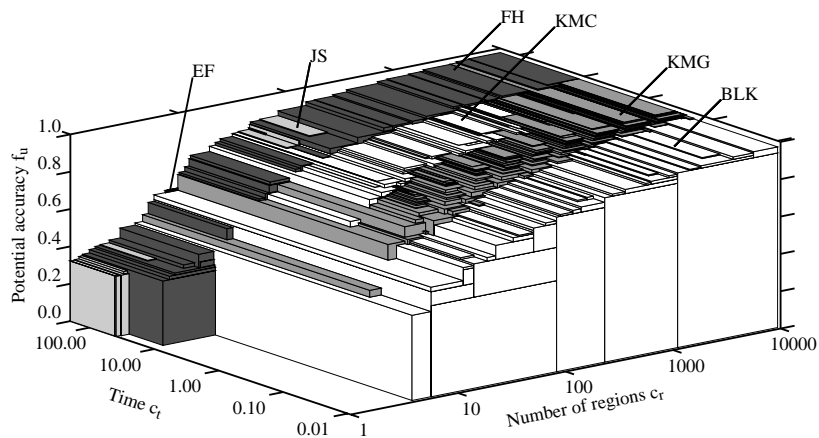


Figure 5: Monotonic hull $f_u \times c_r \times c_t$

are present on the hull, with JS and EF making only a token appearance due to their high processing time relative to accuracy. From this hull we can see that FH gives the highest potential accuracy but if low processing time is a requirement then algorithms KMC and KMG are potentially good choices, obtaining comparable potential accuracy to FH for much shorter processing time, with KMC more accurate than KMG for moderate numbers of regions. If very high potential accuracy is required then all algorithms generate very many regions and interestingly have little advantage over the static segmentation of BLK, which has the advantage of zero processing time. This suggests that image interpretation strategies avoiding a segmentation stage altogether may be worth investigating further.

5 Discussion and conclusions

We have described methods for evaluating segmentation algorithms by points in a joint fitness/cost space, where a point represents a particular instantiation of an algorithm by its parameters. We introduced the “monotonic hull” as an analysis tool which allows us to establish which algorithms perform best with respect to any unknown monotonic fitness function.

In these experiments, depending on the choice of fitness/cost functions different algorithms perform better than others. If little over-segmentation can be tolerated then FH or EF perform most accurately with respect to a pixel-based metric, or KMC with respect to an object proportion metric. If highest potential accuracy is required then FH is the clear winner. Overall, FH performs near-optimally of these algorithms for all metrics. This is an appealing result given its strong theoretical basis [7]. In other experiments not reported here [6] these results were replicated, where FH surpassed other algorithms even on highly textured images despite it not explicitly using texture features. However, FH does have considerably higher processing time requirements than other comparable algorithms. We note also that all algorithms required very high numbers of regions to achieve high potential accuracy. We believe that current algorithms may be approaching the limit of accuracy that can be obtained on real images using solely low-level image

features. Therefore we may have to consider further supervised segmentation algorithms which have knowledge of object classes in order to obtain higher accuracy.

Although ultimately we can never establish the effectiveness of segmentation algorithms except in the context of a full system, we believe that our formulation of the evaluation problem goes some way to being able to obtain meaningful and more general evaluations than previously achievable. Further work remains to be done on characterizing more accurately the costs of what subsequent processing must be applied to a segmented image. Though we have discussed image segmentation alone, we believe that the general framework of analysis using the monotonic hull could usefully be applied to other algorithmic problems.

References

- [1] C. Bradford Barber, D. P. Dobkin, and H. Huhdanpaa. The Quickhull algorithm for convex hulls. *ACM Trans. Mathematical Software*, 22:469–483, 1996.
- [2] S. Belongie, C. Carson, H. Greenspan, and J. Malik. Color- and texture-based image segmentation using EM and its application to content-based image retrieval. In *Proc. ICCV98*, pages 675–682, 1998.
- [3] C. M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, 1995.
- [4] N. W. Campbell, W. P. J. Mackeown, B. T. Thomas, and T. Troscianko. Interpreting image databases by region classification. *Pattern Recognition*, 30:555–563, 1997.
- [5] Y. Deng, B. S. Manjunath, and H. Shin. Color image segmentation. In *Proc. CVPR99*, pages 446–451, 1999.
- [6] M. R. Everingham, H. Muller, and B. T. Thomas. Evaluation of image segmentation algorithms using the Monotonic Hull. Technical report, Department of Computer Science, University of Bristol, 2001. (in preparation).
- [7] P. Felzenszwalb and D. Huttenlocher. Image segmentation using local variation. In *Proc. CVPR98*, pages 98–104, 1998.
- [8] A. K. Jain and F. Farrokhnia. Unsupervised texture segmentation using Gabor filters. *Pattern Recognition*, 24:1167–1186, 1991.
- [9] I. Y. Kim and Hyun S. Yang. An integrated approach for scene understanding based on Markov random field model. *Pattern Recognition*, 28:1887–1897, 1995.
- [10] M. D. Levine and A. Nazif. Dynamic measurement of computer generated image segmentations. *IEEE Trans. PAMI*, 7:155–164, 1985.
- [11] W. Y. Ma and B. S. Manjunath. EdgeFlow: A technique for boundary detection and segmentation. *IEEE Trans. IP*, 2000. (accepted for publication).
- [12] F. Provost and T. Fawcett. Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In *Proc. KDD97*, 1997.
- [13] W. A. Yasnoff, J. K. Mui, and J. W. Bacus. Error measures for scene segmentation. *Pattern Recognition*, 9:217–231, 1977.
- [14] Y. J. Zhang. A survey on evaluation methods for image segmentation. *Pattern Recognition*, 29:1335–1346, 1996.
- [15] Y. J. Zhang and J. J. Gerbrands. Objective and quantitative segmentation evaluation and comparison. *Signal Processing*, 39:43–54, 1994.