Evaluating Image Segmentation Algorithms Using the Pareto Front

Mark Everingham, Henk Muller, and Barry Thomas

Department of Computer Science, 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 the use of the Pareto front to allow evaluation and comparison of image segmentation algorithms in multi-dimensional fitness spaces, in a manner somewhat analogous to the use of receiver operating characteristic curves in binary classification problems. The principle advantage of this approach is that it avoids the need to aggregate metrics capturing multiple objectives into a single metric, and thus allows trade-offs between multiple aspects of algorithm behavior to be assessed. This is in contrast to previous approaches which have tended to use a single measure of "goodness", or discrepancy to ground truth data. We define the Pareto front in the context of algorithm evaluation, propose several fitness measures for image segmentation, and use a genetic algorithm for multi-objective optimization to explore the set of algorithms, parameters, and corresponding points in fitness space which lie on the front. Experimental results are presented for six general-purpose image segmentation algorithms, including several which may be considered state-of-the-art.

Keywords: Vision systems engineering & evaluation; Image segmentation; Multi-objective evaluation; Pareto front.

1 Introduction

Image segmentation is the first stage of processing in many practical computer vision systems. Applications include image interpretation [1,2] and content-based image retrieval [3]. 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 evaluating image segmentation algorithms stems from the ill-defined nature of the problem being addressed. In his survey on evaluation methods for image segmentation [4], 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)." [4]

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 [4] 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 [5] used 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 [6] 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 [6], 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 [7].

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, or degree of over-segmentation (where multiple regions intersect a single object) versus under-segmentation (where a single region intersects multiple objects). Measures which are based solely on a quality or discrepancy 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 [4], but can be seen as a unification of all categories into a consistent framework, defining a general comparison method 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 aggregate fitness function

$$H(a_{\boldsymbol{p}}, I) = \Phi(f_1(a_{\boldsymbol{p}}, I), \ldots, f_N(a_{\boldsymbol{p}}, I)) \quad (1)$$

where $a_{\boldsymbol{p}}$ is a segmentation algorithm a with parameters $\boldsymbol{p}$, and I is a set of ground truth images. Functions $f_i(a_{\boldsymbol{p}}, I)$ are individual fitness functions defined to increase monotonically with the fitness of some particular aspect of the algorithm's behavior. We will also refer to *cost* functions $c_i(a_{\boldsymbol{p}}, I)$, where a cost function can always be trivially derived by negation of a corresponding fitness function.

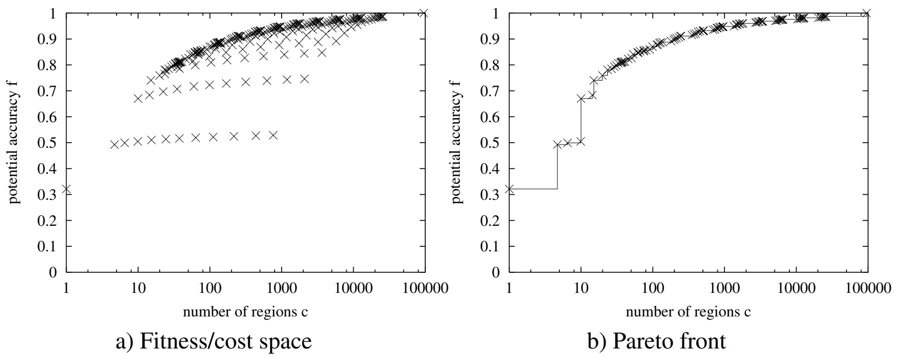

Fig. 1. Example Fitness/Cost Graph and Pareto Front

$\mathit{\Phi}$ aggregates the individual fitness measures into an overall measure of fitness. For example, if $f_1(a_{\boldsymbol{p}}, I)$ measures the accuracy of the segmentation and $c_2(a_{\boldsymbol{p}}, I)$ the execution time of the algorithm, an appropriate form for $\mathit{\Phi}$ might be some nonlinear weighted sum of the two. We believe that in general it is difficult to specify an exact form for $\mathit{\Phi}$, 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 $\mathit{\Phi}$ increases monotonically with increasing values of all fitness functions $f_1, \ldots, f_n$.

With $\mathit{\Phi}$ thus unconstrained, the behavior of an algorithm a applied to a set of images I with a particular choice of parameters $\boldsymbol{p}$ can be characterized by a point in the n-dimensional space defined by evaluation of the fitness functions. Varying the parameters $\boldsymbol{p}$ of the algorithm a produces a new point in the space. By this representation we decouple the fitness/cost trade-off from the particular parameter set used by an individual algorithm. This makes it possible to compare algorithms having different parameter sets.

We can plot a projection of $\mathit{\Phi}$ onto a pair of fitness/cost functions for different parameters $\boldsymbol{p}$ 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 $\boldsymbol{p}$, 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 functions in more detail in Section 5.

3.1 Pareto Front

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, and are said to be "dominated" by another point. In the 2-D case shown, these undesirable parameter settings are points which are to the bottom-right of another point. Since we have defined our aggregate function $\mathit{\Phi}$ to be monotonic, then for a particular data set I and algorithm a parameterized in the space P_a, it is readily shown that the only worthwhile choices of parameters, and corresponding points in fitness space, are in the set

$$\{\langle \boldsymbol{a_p} \in P_a, F(a_{\boldsymbol{p}}, I)\rangle \mid \neg\exists \boldsymbol{a_q} \in P_a : F(a_{\boldsymbol{q}}, I) > F(a_{\boldsymbol{p}}, I)\} \tag{2}$$

where P_a is the parameter space of algorithm a, F is a vector of fitness functions $\langle f_1, \ldots, f_n\rangle$, and a partial ordering relation on F is defined as

$$F(a_{\boldsymbol{q}}, I) > F(a_{\boldsymbol{p}}, I) \iff \forall i : f_i(a_{\boldsymbol{q}}, I) \geq f_i(a_{\boldsymbol{p}}, I) \wedge \exists i : f_i(a_{\boldsymbol{q}}, I) > f_i(a_{\boldsymbol{p}}, I) \tag{3}$$

This construction, originally conceived in the context of political economics by Pareto [8], is referred to as the "Pareto Front". Figure 1b shows points in fitness/cost space on the Pareto front for the graph of Figure 1a, where we have drawn connecting lines to indicate the partitioning of the space by the front.

The strength of the Pareto front is that we can readily show that for *any* choice of aggregate fitness function $\varPhi$ that increases monotonically in F, parameters which do not correspond to points on the Pareto front are *always* bad choices, since there is another point on the front which increases the aggregate fitness.

We can naturally extend our definition of the Pareto front (Equation 2) to the case of multiple algorithms, to obtain the front over both algorithms and their parameters. This has the natural consequence that any algorithm which does not contribute to the front is similarly a bad choice for *any* monotonic fitness function $\varPhi$. An example is shown in Figure 2.

3.2 Convex Hull

The only limitation we have imposed on the form of the aggregate fitness function $\varPhi$ is that it be monotonic. If we can define that $\varPhi$ is a *linear* function of the fitness functions

$$\varPhi(f_1(a_{\boldsymbol{p}}, I), \ldots, f_n(a_{\boldsymbol{p}}, I)) = \sum_{i=1}^{n} w_i f_i(a_{\boldsymbol{p}}, I) + k \tag{4}$$

where without loss of generality $\forall i : w_i \geq 0$, then we can readily show that all worthwhile algorithms and parameters fall on the n-dimensional convex hull [9]. Known parameters w of the linear aggregate function define an iso-fitness hyper-plane through the n-dimensional fitness space, such that all points on that plane have equal aggregate fitness. For example, in the 2-D case of Figure 1, a fitness function $\varPhi = wf(a_{\boldsymbol{p}}, I) + k$ would define a horizontal iso-fitness line such that all algorithms resulting in equal potential accuracy have equal aggregate fitness, regardless of the number of regions output.

The 2-D convex hull has been used in the analysis of receiver operating characteristic (ROC) curves for binary classifiers [10]. 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, and therefore restrict our attention to the more general Pareto front.

4 Approximating the Pareto Front

Determining membership of the Pareto front requires exploring the possibly high-dimensional space of algorithm parameters, and evaluating each fitness function with respect to a test data set. In general, for continuous valued parameter sets, the Pareto front may have infinite cardinality, so in practice we must approximate the front with a finite number of points. Since evaluating the fitness functions is typically computationally expensive, because it requires running the segmentation algorithm on a set of test images, we desire that the front be well approximated with only a moderate number of required fitness evaluations.

In initial experiments [11] we used semi-exhaustive search of the algorithm parameter space to approximate the Pareto front, discretizing the parameter space heuristically. This approach is computationally expensive, and may give a poor approximation of the front because of the high degree of nonlinearity between parameter and fitness spaces. We propose here to instead use a genetic algorithm approach to approximate the Pareto front. This importantly extends the applicability of our method to algorithms with more than two or three parameters, and requires less prior knowledge about the behavior of an algorithm with respect to its parameters.

4.1 PESA Algorithm

The algorithm we use to approximate the Pareto front is a modification of the "Pareto Envelope-based Selection Algorithm" (PESA) proposed by Corne et. al. [12], and shown in Algorithm 1. PESA is a variant of a genetic algorithm, which maintains two populations of chromosomes, with each chromosome encoding an algorithm parameterization. The *internal* population IP contains a set of P_I algorithm parameterizations to be evaluated, and the *external* population EP contains the partial Pareto front of maximum cardinality P_E found thus far in the computation. The principle is to create new algorithm parameterizations for evaluation by exploring areas of fitness space which are poorly represented in the Pareto front found thus far. This is achieved, in steps 11 and 13 (Algorithm 1) by picking parents from which to create new chromosomes which have *low* density in fitness space. Similarly, in step 6 (Algorithm 1), the cardinality of the Pareto front is restricted to a maximum P_E by discarding chromosomes having *high* density in fitness space.

In the original PESA algorithm [12] a form of histogram density estimate was used. We instead use a kernel density estimator [13], with Gaussian kernels having a diagonal covariance matrix. Empirically this has given more stable results in this domain. Other aspects of the algorithm are unchanged: We set $P_I = 10$ and $P_E = 100$, represent algorithm parameters using binary encoding, and use 2-way tournament selection, uniform crossover ($P = 0.7$) and bit-flip mutation ($P = 1/\text{chromosome length}$). The maximum number of fitness evaluations N was set to 1000 in the experiments reported here. More complete details of the algorithm may be found in [12].

Algorithm 1 PESA [12]

```
1: empty external population $EP$
2: generate random internal population $IP$ of size $P_I$
3: **repeat**
4:   evalute $IP$ and add non-dominated members of $IP$ to $EP$
5:   **while** size of $EP > P_E$ **do**
6:     select and remove member of $EP$
7:   **end while**
8:   empty $IP$
9:   **while** size of $IP < P_I$ **do**
10:    **if** probability $P_C$ **then**
11:      select two parents from $EP$ and produce single child by crossover and mutation
12:    **else**
13:      select single parent from $EP$ and produce single child by mutation
14:    **end if**
15:    add child to $IP$
16:  **end while**
17: **until** number of evaluations = $N$
18: return $EP$
```

5 Fitness and Cost Functions

For the experiments reported in Section 6 we defined five fitness/cost functions. Our basis is a set of ground truth images $\{I_1, \ldots, I_N\}$, each of which has been manually segmented into a set of objects O_i according to a pre-defined set of object classes Λ, such that each pixel in an image is assigned a desired object identifier.

Pixel-wise Potential Accuracy. We first define a pixel-wise "potential accuracy" fitness function. Given a segmentation output from a parameterized algorithm $a_{\boldsymbol{p}}$, we assign to each region the object identifier which maximizes the fitness, such that each pixel x is assigned an object identifier $\hat{o}_x$. The fitness is calculated as a mean across images:

$$f_{pa}(a_{\boldsymbol{p}}, I) = \frac{1}{|I|} \sum_{i \in I} \frac{\sum\limits_{o \in O_i} |\{x \in o | \hat{o}_x = o\}|}{\sum\limits_{o \in O_i} |o|} \tag{5}$$

where $|o|$ denotes the size in pixels of object o.

This measures to what extent the desired object identifier output can be achieved given the segmentation, assuming a perfect classifier. It can be interpreted as a measure inversely proportional to the degree of *under-segmentation*.

Object-wise Potential Accuracy. A potential problem with the use of pixel-based metrics is that they exhibit significant bias if the test images have a skewed distribution of object sizes. Small objects, or classes of object which tend to appear small in the image, contribute little to the metric. We therefore also consider a potential accuracy function normalized such that all objects and classes of object are defined to be of equal importance:

$$f_{oa}(a_{\boldsymbol{p}}, I) = \frac{\sum\limits_{i \in I} \sum\limits_{o \in O_i} \frac{1}{|o|} |\{x \in o | \hat{o}_x = o\}| w_{oa}(o)}{\sum\limits_{i \in I} \sum\limits_{o \in O_i} w_{oa}(o)} \tag{6}$$

where $w_{oa}(o)$ is a weighting function

$$w_{oa}(o) = \begin{cases} \frac{1}{P(\lambda_o)} & \text{if } |o| \geq k \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

and $P(\lambda_o)$ is the prior probability of the object class $\lambda \in \Lambda$ from which object o is drawn. The constant k removes from consideration objects of very small size. In the experiments reported in Section 6 we set k to approximately 0.05% of the image area.

We can interpret the normalized potential accuracy function as giving a measure of the proportion of each object which can be correctly identified given the segmentation.

Region-wise Information Content. The third fitness function we define measures the proportion of an object which is present in a single region:

$$f_{ri}(a_{\boldsymbol{p}}, I) = \frac{\sum\limits_{i \in I} \sum\limits_{r \in R_i} \frac{1}{|\hat{o}_r|} |\{x \in r | o_x = \hat{o}_r\}| w_{ri}(\hat{o}_r)}{\sum\limits_{i \in I} \sum\limits_{r \in R_i} w_{ri}(\hat{o}_r)} \tag{8}$$

where $\hat{o}_r$ is the optimal object identifier assigned to region r given the segmentation, and $w_{ri}(o)$ is a weighting function

$$w_{ri}(o) = \begin{cases} 1 \text{ if } |o| \geq k \\ 0 \text{ otherwise} \end{cases} \tag{9}$$

If we make the naive assumption that information about an object is uniformly distributed among it's pixels, this function measures what proportion of object information is available within a single region. It can be interpreted as a measure of *over-segmentation*.

Simple Cost Functions. We additionally define two simple cost (negative fitness) functions: c_r measures the mean number of regions per image output by a segmentation algorithm, and is related to the degree of over- or under-segmentation. If we assume constant cost per region, it also characterizes the processing time required by whatever vision system follows the segmentation stage. Finally, c_t measures the mean processing time per image of the segmentation algorithm.

6 Experiments

6.1 Image Data

In the experiments reported here we compared segmentation algorithms over a set of 100 images of urban and rural outdoor scenes taken from the Sowerby image database [14, 15]. These images were hand-segmented, and pixels classified into one of eight high-level object classes, such as road, pavement, vehicle etc. Objects correspond to single instances from these classes, and the desired segmentation is for each object to comprise a single region. Since the objects classes are semantically high-level, this represents a challenging task for a low level image segmentation algorithm.

6.2 Segmentation Algorithms

We ran experiments with six 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. We restricted our experiments here to algorithms using only grey-level or color features.

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 and KMC use a common approach of clustering on low-level features using the K-means algorithm [16], forming connected regions, and merging regions until a minimum region size is obtained. KMG uses grey-level features, while KMC uses intensity and color features in the opponent color space.

FH [17] formulates segmentation as a graph cutting problem and 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 [18] uses color quantization followed by a multi-scale region growing step which aims to segment both uniform and textured regions.

EF [19] uses predictive coding to estimate the direction of change in color features, and forms boundaries by propagating the estimated flow field.

7 Results

7.1 Pixel-Wise Potential Accuracy

Figure 2 shows the Pareto front for all algorithms in the f_{pa} (pixel-wise potential accuracy) $\times$ c_r (number of regions) space. 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 results outside these ranges is unlikely to be useful. Algorithms which have *no* points on the front are shown grayed out in the key, in this case BLK, KMG, and EF.

Overall we can see that around 10 regions per image are needed to achieve 80% mean potential accuracy, with around 1000 regions to reach 90%. The front is broadly divided into three intervals. Algorithm JS gives highest potential accuracy for very small numbers of regions, FH for moderate numbers of regions, and KMC for larger numbers of regions. Interestingly we can achieve over 50% potential accuracy with just 2 regions, indicating the possible bias in the use of pixel-wise metrics. In this case this arises from the predominance of pixels from the "road" class in the Sowerby image database.

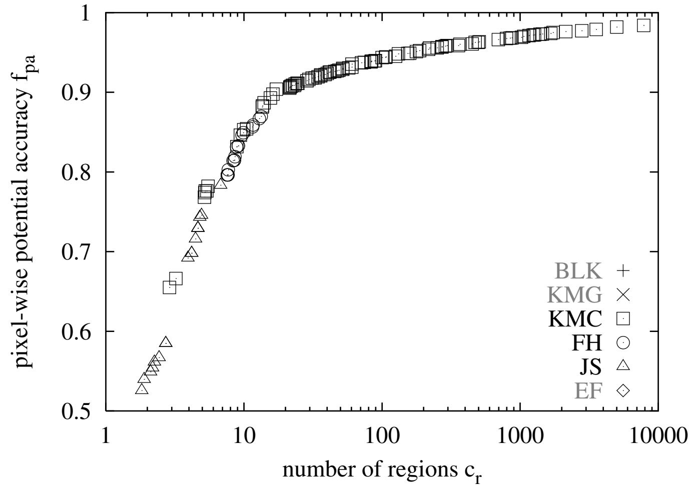

Fig. 2. Pareto Front of Fitness $f_{pa} \times$ Cost c_r

The absence of BLK, KMG, and EF algorithms from the front indicates that the other algorithms result in higher potential accuracy for the same number of regions output. The dominance of the front by the KMC algorithm is interesting, because of the simplicity of this algorithm. In experiments using semi-exhaustive search [11] this algorithm was evaluated poorly, with FH dominating the front. Comparison of the results using semi-exhaustive search and the PESA approach revealed that the improved evaluation of KMC is due to a more accurate approximation of the Pareto front for this algorithm.

7.2 Object-Wise Potential Accuracy

Figure 3 shows the Pareto front for all algorithms in the f_{oa} (object-wise potential accuracy) $\times$ c_r (number of regions) space.

Using the object-wise potential accuracy metric (Equation 6) reveals the bias in the algorithms towards formation of large regions. In this case, where all objects are considered of equal importance, we see that many more regions are required to reach comparable levels of potential accuracy: around 100 regions are required to potentially identify 80% of each object correctly, compared to 10 for 80% of pixels, and around 5,000 regions to reach 95%. The front is dominated by the KMC and FH algorithms, with KMC performing better for fewer regions, and FH for larger numbers of regions. The other algorithms contribute only a few points in the very low or high ranges of number of regions, and EF contributes no points.

7.3 Region-Wise Information Content

Figure 4 shows the Pareto front for all algorithms in the f_{oa} (object-wise potential accuracy) $\times$ f_{ri} (region information content) space.

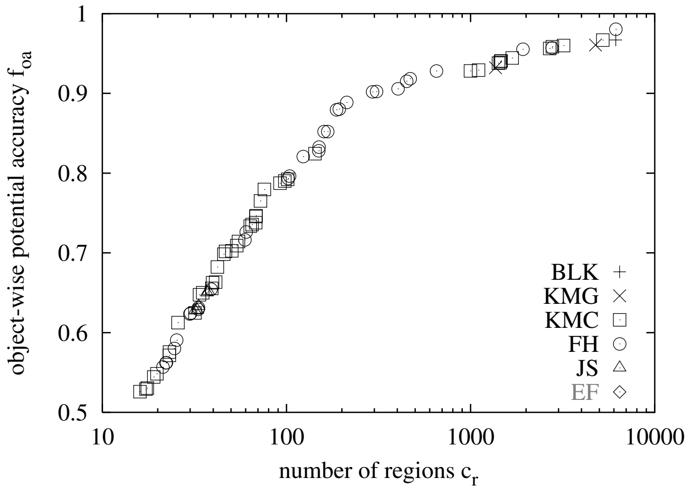

Fig. 3. Pareto Front of Fitness f_{oa} × cost c_r

There is an approximately linear inverse relationship between the object-wise potential accuracy and the proportion of each object available within a single region. In absolute terms, all algorithms over-segment the image quite severely with respect to our choice of segmentation ground truth, perhaps unsurprisingly given the high level nature of our desired segmentation. To obtain 80% potential accuracy by objects, each region contains an average of around 15% of the object pixels. This suggests that object classification based on single regions [2] is likely to be unrealistic for this data set. The front is once again dominated by the KMC and FH algorithms, with other algorithms contributing points only for extreme over- or under-segmentation.

7.4 Processing Time

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_{pa} (pixel-wise potential accuracy) × c_r (number of regions) × c_t (processing time) space. This allows us to evaluate the trade-offs which might be made between potential accuracy, complexity of output, and the processing time required to obtain a segmentation. Figure 5 shows a view of the resulting Pareto front.

All algorithms except EF appear on the front. The EF algorithm produced output with too low potential accuracy relative to it's very high processing time. The JS and FH algorithms also contribute only small areas to the front because of their high processing time. The BLK algorithm occupies a large area because of it's zero processing time (recall that the segmentation in this case is constant and independent of the input image). This may seem surprising, but if we require very high potential accuracy, we observe that none of the algorithms have a clear advantage over static segmentation into blocks, and have of course higher processing time. This result is also interesting in the context of

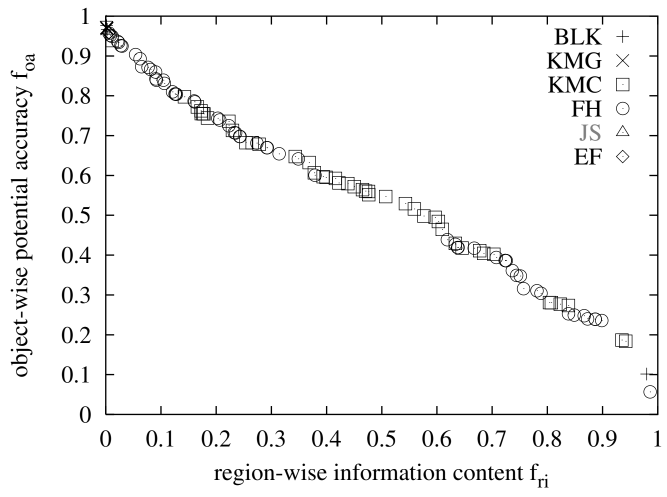

Fig. 4. Pareto Front of Fitness f_{oa} × Fitness f_{ri}

several recent approaches to object detection [20,21,22,23] which avoid a segmentation stage altogether.

For moderate numbers of regions, the KMG and KMC algorithms dominate, offering comparable accuracy to FH, but with lower processing time. These algorithms are two to three orders of magnitude faster than the FH algorithm, and the KMC algorithm in particular gives higher or comparable potential accuracy versus number of regions.

8 Conclusions

In this paper we have described methods for evaluating image segmentation algorithms in multi-dimensional fitness spaces, capturing multiple conflicting objectives of segmentation, and have proposed the use of the Pareto front as an analysis tool which allows us to establish which algorithm or choice of parameters gives best performance with respect to any unknown monotonic fitness function. A modified version of the PESA algorithm [12] allows approximation of the front in reasonable time. We have demonstrated how this approach can yield informative results about practical image segmentation algorithms.

In our experiments, we saw that the choice of the best algorithm for our defined task is dependent on requirements with respect to allowable degree of over- or under-segmentation, and processing time. These trade-offs are straightforward to assess and visualize using the Pareto front. On our particular task, the KMC algorithm, using K-means clustering of color features, seems the best choice, giving comparable results to the FH algorithm, but one to two orders of magnitude faster. This result is interesting to us, since this ordering of algorithms only became clear as a result of introducing the PESA approximation of the Pareto front, which yielded more accurate approximation than heuristic-guided semi-exhaustive search. It is also interesting because of the simplicity of the KMC algorithm. The FH algorithm [17], which performed comparably, has a

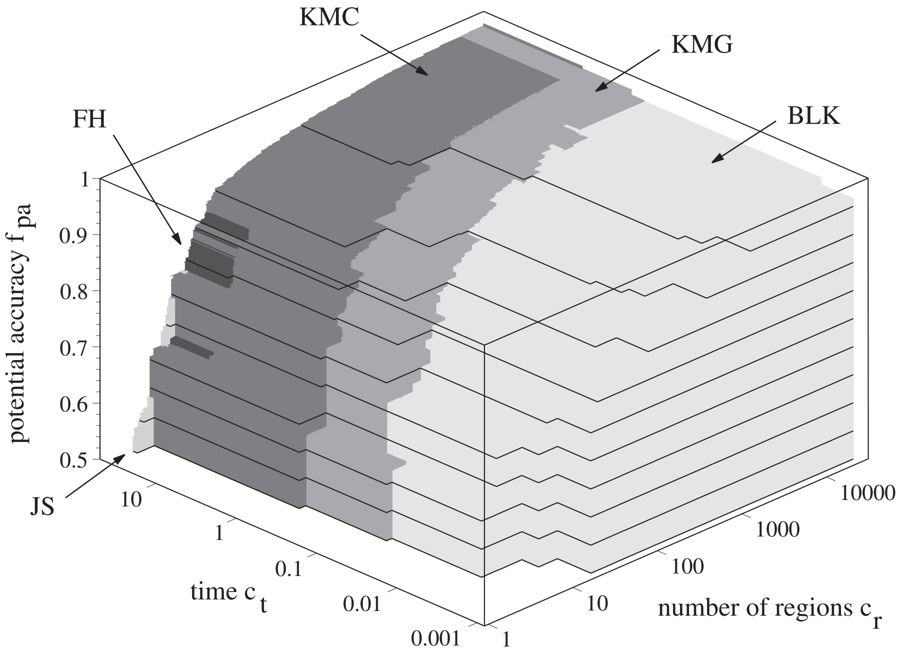

Fig. 5. Pareto Front of $f_{pa} \times c_r \times c_t$

strong basis in terms of guaranteeing neither over- nor under-segmentation with respect to a color edge strength metric, but we conjecture that this metric is too weak for the optimality of the overall approach to yield significant benefits.

We note also that all algorithms required very high numbers of regions to achieve high potential accuracy on our task. While our desired segmentation is admittedly very high-level, we have observed the same behavior with lower-level "parts-based" segmentations. 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, or stronger forms of image prior. Comparison of the pixel-wise and object-wise metrics shows that the commonly used heuristic of requiring output regions to be of some minimum area may not be a generally useful heuristic.

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 using the Pareto front goes some way to being able to obtain informative and more general evaluations than previously achievable, while retaining a formal framework. In our work on image classification in the domain of the Sowerby image database [14,15] we have used our evaluation method to select a more effective segmentation algorithm than previously used [2] and improved classification accuracy by around 8% [24].

Further work remains to be done on defining informative fitness functions for image segmentation, for example characterizing more accurately the costs of what subsequent processing must be applied to a segmented image. If we are to be able to compare published image segmentation algorithms with any certainty, we also need to specify more strictly the desired image segmentation within a particular domain, and produce appropriate ground truth data. We also note that the research area of multi-objective optimization may yield more efficient and accurate methods for approximation of the Pareto front.

Acknowledgements. This work was partially sponsored by Hewlett-Packard Research Laboratories, Bristol, UK. This support is gratefully acknowledged.

References

1. Kim, I.Y., Yang, H.S.: An integrated approach for scene understanding based on Markov random field model. Pattern Recognition **28** (1995) 1887–1897
2. Campbell, N.W., Mackeown, W.P.J., Thomas, B.T., Troscianko, T.: Interpreting image databases by region classification. Pattern Recognition (Special Edition on Image Databases) **30** (1997) 555–563
3. Belongie, S., Carson, C., Greenspan, H., Malik, J.: Color- and texture-based image segmentation using EM and its application to content-based image retrieval. In: Proceedings of IEEE Conference on Computer Vision (ICCV98). (1998) 675–682
4. Zhang, Y.J.: A survey on evaluation mehods for image segmentation. Pattern Recognition **29** (1996) 1335–1346
5. Levine, M.D., Nazif, A.: Dynamic measurement of computer generated image segmentations. IEEE Transactions on Pattern Analysis and Machine Intelligence **7** (1985) 155–164
6. Yasnoff, W.A., Mui, J.K., Bacus, J.W.: Error measures for scene segmentation. Pattern Recognition **9** (1977) 217–231
7. Zhang, Y.J., Gerbrands, J.J.: Objective and quantitative segmentation evaluation and comparison. Signal Processing **39** (1994) 43–54
8. Pareto, V.: Manual of Political Economy. A. M. Kelley, New York (1971) Original French 1906.
9. Barber, C.B., Dobkin, D.P., Huhdanpaa, H.: The Quickhull algorithm for convex hulls. ACM Transactions on Mathematical Software **22** (1996) 469–483
10. Provost, F., Fawcett, T.: Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In Heckerman, D., Mannila, H., Pregibon, D., Uthurusamy, R., eds.: Proceedings of International Conference on Knowledge Discovery and Data Mining (KDD-97), AAAI Press (1997)
11. Everingham, M.R., Muller, H., Thomas, B.T.: Evaluating image segmentation algorithms using monotonic hulls in fitness/cost space. In: Proceedings of British Machine Vision Conference (BMVC2001). (2001) 363–372
12. Corne, D.W., Knowles, J.D.: The Pareto-envelope based selection algorithm for multiobjective optimization. In: Proceedings of International Conference on Parallel Problem Solving from Nature (PPSN VI). (2000) 839–848
13. Silverman, B.W.: Density Estimation for Statistics and Data Analysis. Chapman and Hall (1986)

14. Mackeown, W.P.J.: A Labelled Image Database and its Application to Outdoor Scene Analysis. PhD thesis, University of Bristol (1994)
15. Collins, D., Wright, W.A., Greenway, P.: The Sowerby image database. In: Proceedings of IEE Conference on Image Processing and its Applications (IPA99). (1999) 306–309
16. Bishop, C.M.: Neural Networks for Pattern Recognition. Clarendon Press, Oxford (1995)
17. Felzenszwalb, P., Huttenlocher, D.: Image segmentation using local variation. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR98). (1998) 98–104
18. Deng, Y., Manjunath, B.S., Shin, H.: Color image segmentation. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR99). (1999) 446–451
19. Ma, W.Y., Manjunath, B.S.: EdgeFlow: A technique for boundary detection and segmentation. IEEE Transactions on Image Processing **9** (2000) 1375–1388
20. Schiele, B., Pentland, A.: Probabilistic object recognition and localization. In: Proceedings of IEEE Conference on Computer Vision (ICCV99). (1999) 177–182
21. Papageorgiou, C., Poggio, T.: A pattern classification approach to dynamical object detection. In: Proceedings of IEEE Conference on Computer Vision (ICCV99). (1999) 1223–1228
22. Yang, M.H., Roth, D., Ahuja, N.: A snow-based face detector. In: Advances in Neural Information Processing Systems 12. MIT Press (2000) 855–861
23. Weber, M., Welling, M., Perona, P.: Unsupervised learning of models for visual object class recognition. In: Proceedings of European Conference on Computer Vision (ECCV00). (2000)
24. Everingham, M.: Methods and Metrics for Image Labelling with Application to Low Vision. PhD thesis, Department of Computer Science, University of Bristol (2002)