

TIME-WEIGHTED EVALUATION OF IMAGE SEGMENTATION WITH A GENETIC ALGORITHM

Hassan Almuhairei, Martin Fleury

School of Comput. Sci. & Elec. Eng., University of Essex, Colchester, United Kingdom
hoalmu@essex.ac.uk, fleum@essex.ac.uk

Adrian F. Clark

School of Comput. Sci. & Elec. Eng., University of Essex, Colchester, United Kingdom
alien@essex.ac.uk

Keywords: Image segmentation, Genetic algorithm, Quantitative evaluation

Abstract: Quantitative evaluation of segmentation algorithms implies rigorous testing against ground truth segmentations. Though under-reported in the literature, the performance of a segmentation algorithm depends on the choice of input parameters. When testing extends over an algorithms parameter space, then the search for satisfactory settings has a considerable cost in time. This paper considers the use of a genetic algorithm (GA) to avoid an exhaustive search. As application of the GA drastically reduces search times, the paper investigates how best to apply the GA in terms of initial candidate population, convergence speed, and application of a final polishing round. To further reduce evaluation time and subsequent image batch-processing times, this paper introduces a time factor into the GA cost function. It is found that this procedure while preserving the GA solution also improves interpretation and parameter selection.

1 INTRODUCTION

Contrasting image segmentation to recognition tasks such as the use of handwriting, and face databases, the authors of (Martin et al., 2001) remark “Typically [in segmentation] researchers will show their results on a few images and point out why the results ‘look good’”. Part of the problem may be the logistics of quantitative evaluation in performing a large number of evaluations, as an exhaustive search with multiple parameter settings is an onerous task. Consequently, we have used a genetic algorithm (GA) (Goldberg, 1989) search module in our evaluation environment to decrease the processing time for the search as a whole.

The GA acts to optimize the selection of parameters. As is well-known, a GA emulates to some extent the supposed process of genetic evolutionary adaptation. While in genetics a chromosome is a molecular package containing genes, in a GA a chromosome becomes a vector containing a set of variable parameters. The GA employed was a real-valued GA inspired by Polheim’s GEATbx, a GA toolbox for Matlab (Polheim, 2005) (though the output may be integer-valued parameter settings). It employs ex-

tended intermediate recombination (Mühlenbein and Schlierkamp-Voosen, 1993), wherein an offspring gene g_O is conceived from its two parent genes g_1 and g_2 by

$$g_O = \alpha g_1 + (1 - \alpha) g_2 \quad (1)$$

where α is a scaling factor in the range $[-d, 1 + d]$. Assuming that the N genes in a chromosome form an N -dimensional hypercube then, when $d = 0$, g_O lies along the straight line between g_1 and g_2 ; if $d > 0$, extrapolation is permitted, though in this work, $d = 0$. The algorithm employed allows mutation of variable parameter values as a means of preventing chromosomes becoming too close to each other and remaining in local minima.

Choice of algorithm parameters can critically determine the final segmentation output. Selection of parameters can occur in two ways: 1) algorithms that automatically determine parameter settings “on-line”; and 2) the class of algorithms now considered which need to be tested “off-line” (Zhang et al., 2005). Given the range of parameter values, the starting point is to select random values for each parameter. As there can be only one solution to choice of segmentation parameters, the best chromosome is chosen at the end of a run according to its cost function value.

It is possible to select a stopping point by checking when the difference in the cost or fitness function's value between successive generations passed below a threshold. However, because of the relatively lengthy time taken to evaluate each application of a segmentation algorithm to an image, stopping after a given number of generations is a practical alternative. The rate of convergence towards a global minimum value of a cost function to assess an algorithm was found to be determined by the size of the population at each generation. However, the convergence over many generations is also important.

The main contribution of this paper is adding the time taken to complete the segmentation for each parameter set as a factor in the cost function. The rationale behind this addition is that the quality of the segmentation results is not the only value one would like to improve, as there is also a need to balance the quality with the segmentation processing time. Adding this time factor gave some insight into the significance of some of the algorithm parameters not only in respect to the processing time performance but also to the quality of the segmentation performance. For instance, while experimenting without using the time factor, the GA module will randomly vary certain parameters that actually do not affect the overall quality of the segmentation, once optimization of the significant parameters for the algorithm has taken place.

However, this parameter variation can have a great effect on the processing time performance. Therefore, adding the time factor can still optimize the significant parameters while giving a better computational performance. There is another advantage to adding a time factor, as it not only optimizes the parameters for the segmentation algorithm itself according to the processing time performance, but also optimizes the whole evaluation process, because the GA module is inclined to choose parameters that are time efficient. As a consequence, the whole evaluation process takes less processing time to complete if quantitative assessment is applied.

Increasingly (Martin et al., 2001) the importance of systematic validation on sufficient datasets has been realized, the quantitative testing approach. For a given input, the corresponding correct output is also known, either because the input is simulated or by virtue of either expert opinion or the presence of a ground truth such as hand-segmented images. Implicit in this procedure is that the finite set of test data is representative of variation in the real world and that the number of samples is large enough (Guyon et al., 1998). A number of sources suitable for quantitative testing exist. At the University of Berkeley, CA Martin (Martin et al., 2001) supervised the

creation of a database of 12,000 hand-labeled (from a pool of 30 human subjects) segmentations of images taken from the Corel dataset of 1,000 images. The smaller Sowerby database (Rees and Greenway, 1999) is an earlier collection of hand-segmented images which has been used to evaluate edge-detection algorithms (Konishi et al., 1995). The work in (Borra and Sarkar, 1997) also used ground truth from a set of fifty images and applied analysis of variance to five evaluation metrics for object recognition algorithms. The Berkeley database encourages users to download benchmarking code as well as 200 training images and a further 100 test images of size 240×160 pixels. Therefore, we have exploited this database to select images and their ground truths for the evaluation process.

The paper now considers in Section 2 applying a GA to segmentation without a time factor before seeing what the effect of factoring in time is, Section 3. Finally, Section 4 draws some conclusions.

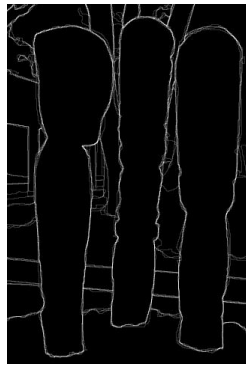
2 OPTIMIZING PARAMETERS WITH A GA

GAs (and other machine learning techniques) have been previously employed in computer vision and image processing (Olague, 2007) in general and in particular image segmentation (Chabrier et al., 2005). Additionally, there have been some studies on formulating image segmentation as an optimization problem (Levachkine and Sossa, 2000; Schoenemann and Cremers, 2007). Intuitively, the case of the parameter-fitting for image segmentation also inherently lends itself as an optimization problem. GAs by definition are designed to achieve good results in those types of problem that have a wide solution space. The idea of factoring in time was initially introduced in Everingham et. al. (Everingham et al., 2001). The analysis of the importance of considering time as a factor in segmentation is interesting. However, that work did not discuss in detail what type of parameters were used and varied for each segmentation algorithm.

The mean-shift algorithm (Comaniciu and Meer, 2002) makes a convenient example, especially as the authors have made EDISON code available at <http://www.caip.rutgers.edu/riul/research/robust.html>, for which we are grateful. Fig 1a is a test image from the Berkeley database, Fig 1b is an example hand-segmentation also included in the database. Fig. 1c shows the result of varying the mean-shift parameters. Higher values of `radiusR` results in less regions, while higher values of `radiusS`



a



b

radiusS	1	10	1	10
	1	1	10	10
Colour Distance	1	1	10	10
1				
10				

c

Figure 1: (a) Example image (Easter island statues) from Berkeley segmentation database (b) human hand-segmentation (c) variation of segmentations with parameter settings.

effectively results in more computation but smoother region boundaries. The GA of Section 1 was now applied to this image.

Table 1 is an illustrative rather than representative run showing the generation-by-generation most fit selection of ‘chromosomes’ for Fig. 1a. Three parameters formed the chromosomes: `radiusR` (the range radius of the mean-shift sphere in color space), `radiusS` (the spatial radius in grid space), and `colorDistance` (defining the region merge threshold). There was a population of just five for each generation, which explains why the cost function’s value does not reach a minimal value (as shown next). Two GA parameters, the recombination rate, *i.e.* α in the line recombination, and the mutation rate were set to 0.6 and 0.8 respectively. The latter governs the ability to escape from a local minimum. As with such evolutionary algorithms, it is not otherwise possible to directly govern their behavior, which is probably their principal disadvantage. However, the reduction in duration at reaching a selection adequately compensates for that.

A disadvantage of truncating the search after a fixed number of generations is that there is no guarantee that the GA could be (say) performing hill climbing to leave a suspected local minimum. In Table 1, the seventh generation value is less than the eighth and, therefore, had the search been concluded at the eighth unhelpful parameter values would be selected. A possible solution to this problem is to include a refinement or polishing stage in which the path taken by the GA over each generation is inspected to more closely approach a global minimum. This issue is returned to at the end of this Section in relation to graph-based segmentation. Experimenting with the GA as in

Table 1: Trial evaluation of mean-shift parameters with the GA.

		Chromosomes		
Gen.	Cost	radiusS	radiusR	colorDistance
1	18.63	3	8	16
2	14.17	3	14	16
3	15.27	3	16	14
4	14.61	1	16	17
5	16.74	11	4	17
6	13.76	11	18	19
7	11.87	3	17	19
8	14.26	12	17	19
9	11.19	2	15	19

Table 2 with a population of ten chromosomes in each generation gives rise to interesting observations. The first point to notice is that the cost function’s value remains the same throughout the tests. The rapid descent to a global minimum arises because a population of ten was taken for each generation. The other important point is that although `radiusR` and `radiusS` do not converge to produce an optimal set of parameters, `colorDistance` is never lower than 11. We found that these two points are always true for the mean-shift algorithm over a range of different images, and although the lowest score reached will be different for each image, it can be concluded that `radiusR` and `radiusS` do not make a significant difference to the result as long as the `colorDistance` parameter is larger than eight.

The more common alternative to truncating the search after a fixed number of generations with a large initial population is to complete the search after a con-

Table 2: Trial output of mean-shift parameters with the GA.

Gen.	Cost	Chromosomes		
		radiusS	radiusR	colorDistance
1	4.71	2	5	16
2	4.71	2	16	11
3	4.71	2	16	11
4	4.71	2	4	17
5	4.71	2	2	17
6	4.71	16	2	19
7	4.71	2	10	16
8	4.71	2	18	15
9	4.71	2	16	12
10	4.71	2	17	12
11	4.71	2	10	19
12	4.71	2	10	19
13	4.71	2	10	17
14	4.71	2	15	15
15	4.71	3	11	16
16	4.71	3	18	19
18	4.71	4	15	19
19	4.71	8	2	19
20	4.71	1	3	19

vergence criterion had been met. How the search is conducted is a pragmatic decision, as in essence a GA is simply a more disciplined way than random probing to explore a large problem space. To examine the behavior over successive generations a very low population of just two members was deliberately chosen to give slow convergence. The recombination rate was set to 0.6 and the mutation rate was set to 0.2 for the mean-shift segmentation parameters. In Fig. 2, a best-fit linear regression line is found by a standard numerical method, without any claims for goodness-of-fit. The line shows that the cost function values have a slight negative trend showing continuous improvement, though with significant divergences and even lower values of the cost function at around 40 generations. In Fig. 3, the values of the contributory parameters are superimposed. The values of the cost function are connected by a moving average line, with the average taken over three data points. In broad terms, the behavior of the values chosen by the GA for the other parameters is oscillatory, displaying what is close to being a systematic sampling of the parameter space.

Graph-based segmentation (Felzenszwalb and Huttenlocher, 2004) also displayed a similar dependence on choice of parameters. We gratefully acknowledge source code from <http://people.cs.uchicago.edu/~pff/segment/>. There are three input parameters available for this

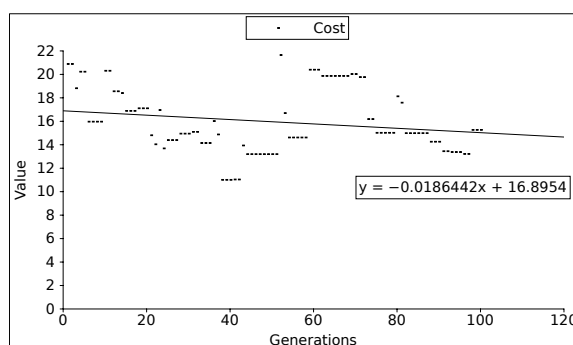


Figure 2: Mean-shift parameter convergence with linear trend for 100 generations of the cost function value.

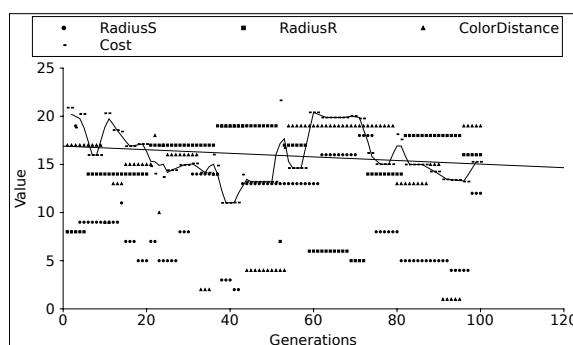


Figure 3: Mean-shift parameter convergence with linear trend of the cost function value, including the behavior of contributory parameters

algorithm: 1) σ (sigma), the variance of a Gaussian smoothing filter prior to segmentation; 2) k , which sets the scale of the components found through a thresholding function; and 3) Min , which is the minimum component size enforced by post-processing. Fig. 4 then shows the considerable variety of segmentations arising from these parameter settings for the same original image as Fig. 1. Similarly to Fig. 3 and with the same GA search parameters, Fig. 5 shows 100 generations of a parameter search, again plotting a moving average through the data points. The slight negative of the linear trend is more apparent in this representation, as is the oscillatory nature of the testing of individual parameters. As mentioned earlier, it is possible to refine the output of the GA by application of a non-GA polishing algorithm to what is a non-constrained, non-linear optimization problem. Newtonian methods are unsuitable if the cost function to be optimized is non-differentiable. Therefore, this work used the Nelder-Mead direct search, ‘simplex method’ (Nelder and Mead, 1965). In the implementation, previous values found by the GA form the input to the algorithm along with the cost function. The values form the vertices of the

Sigma \ K	Min	20	2000	20	2000
	K	50	50	500	500
0.1					
1					

Figure 4: Variation of segmentations of Fig. 1 with parameter settings for graph-based segmentation (Felzenszwalb and Huttenlocher, 2004)

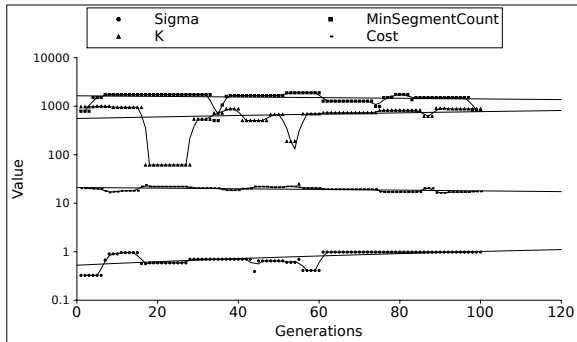


Figure 5: Graph-based segmentation parameter convergence with linear trend of the cost function value, including the behavior of contributory parameters. (Note the logarithmic vertical axis.)

simplex and at each iteration the worst one of these values is replaced. This is achieved by a number of trial evaluations of the cost function at the vertex after the simplex has been reflected, expanded or contracted. Fig. 6 shows how the Nelder-Mead procedure will find a lower value than that given at the end of the GA iterations. However, for this value the parameter settings found are no lower than those of the minimum value found in the course of the GA iterations. Therefore, applying the Nelder-Mead procedure should certainly improve upon the final GA result but the effect is to deepen that result rather than find a global minimum within parameter space.

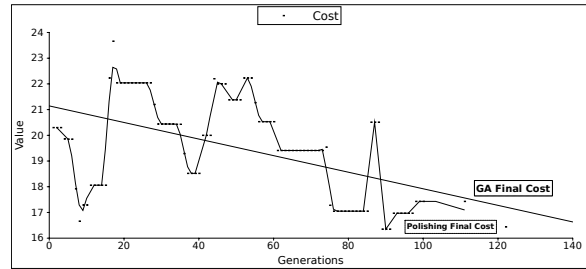


Figure 6: Application of Nelder-Mead polishing to GA optimization, showing the final GA derived value and the value found after further iterations of the polishing algorithm.

3 ADDING TIME AS A FACTOR

Adding processing time to the cost function can take place in various ways such as through an additive or multiplicative factor. Using a multiplicative factor provides a trade-off between segmentation evaluation and computational time, and, therefore, after initial investigations, the cost function was modified in this way. It was decided that including a time factor as an exponential weighting gave too much emphasis to achieving low processing times. The time that the GA itself took for processing was not included, as this time was negligible and certainly less than 5% of any segmentation processing time.

For these experiments, the population size was set to 20 and the first 20 generations were run. The recombination rate was fixed at 0.6 and the mutation rate at 0.2. By observation, the GA module reaches an acceptable stable solution in much fewer generations when a comparatively large population size is employed. To see the start of the stabilization trend in the parameter search with the time factor, Fig. 7 shows the first three generations. The members of the population are plotted across the horizontal axis and the parameter values for a population member can be

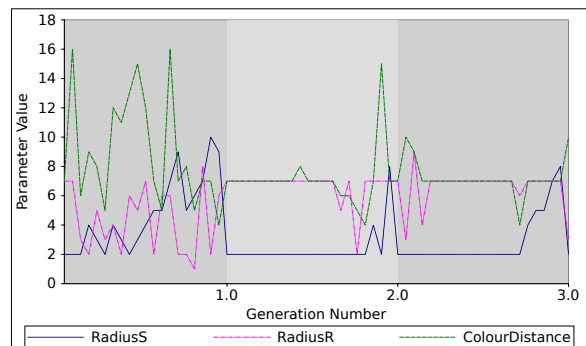


Figure 7: The first three generations of the meanshift GA evaluation

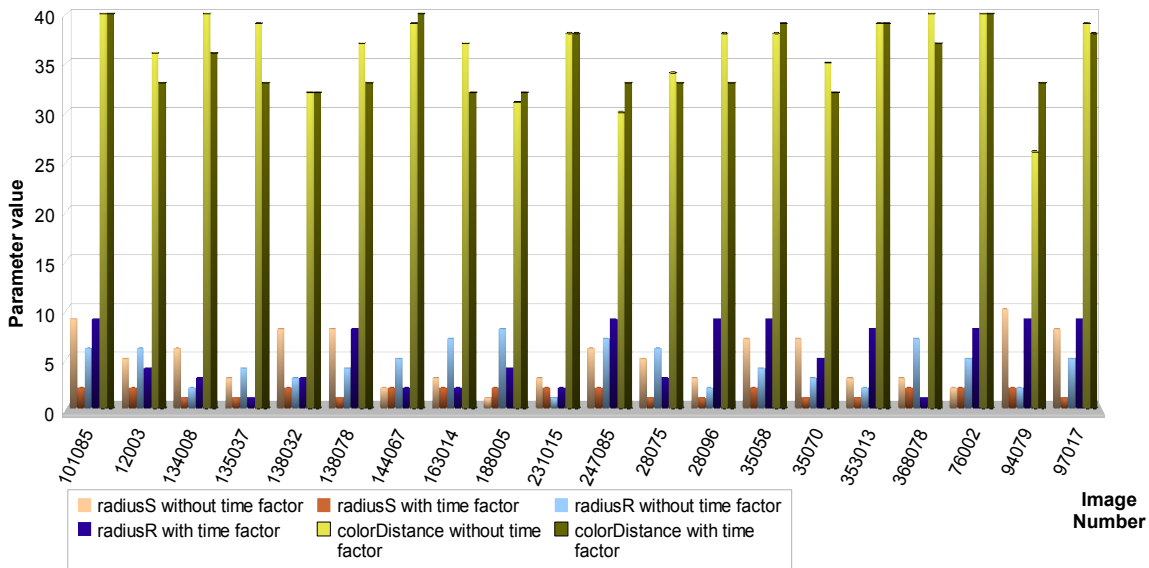


Figure 8: Meanshift segmentation evaluation for 20 images with and without a time factor

read off in the vertical direction. At the crossover point between the generations, the fittest parameter set is shown. It is clear from the parameter variation in the second and the third generations that the selection is already stabilizing. For example RadiusS tends to stabilize at value 2 and RadiusR at value 5. Therefore, employing a time factor arrives at similar results for the example image but may well increase the convergence speed as the values of less significant parameters are explored less. Fig. 8 shows the application of the GA with and without a time factor in the cost function. The horizontal axis is annotated with the image numbers of 20 images from the Berkeley database. Consider the effect of the time factor on the value of the radiusS parameter: when the time factor is present, the value of this parameter is always equal or less than two. While without the presence of the time factor the same parameter value does not have a specific trend, and changes between different images in the test. The best explanation for this is that this parameter does not have a great significance for the quality of the segmentation. However, higher values of this parameter are computationally expensive. There is no similar trend for the radiusR parameter, and the time factor also does not have any noticeable effect on the third colorDistance parameter.

3.1 Watershed segmentation

Experiments with the Watershed algorithm (Vincent and Soille, 1991) also gave rise to a variety of results, depending on choice of parameters. Fig. 9 illustrates

the optimal results found by the evaluation with and without the time factor. The main parameters used were firstly a watershed threshold parameter for the core watershed algorithm. This parameter is varied between 1 and 80. In our tests, a k -mean color quantization stage was also added as a pre-processing stage. The number k here refers to the number of colors that the image will be reduced to. The final parameter considered was the maximal number of iterations parameter for the k -mean algorithm. This is a parameter that controls how many iterations are carried out to search a pixel's neighbors for color similarity as part of the quantization process. The first point to observe is that the threshold arrived at after application of the GA is always very high, higher than 60, and there is no difference in this between using the time factor or not using it. The reason for this is that higher thresholds tend to eliminate smaller details and segments that are not noticed by the human hand-segmenter and as such the evaluation tends to prefer higher values for the threshold parameter. Another point to observe is that the evaluation tries to optimize the parameter set with lower values for the color quantization parameter, which means a more smoothing of the input images. Noting that the maximum value for k is 256, then 50 is a relatively low number and results in high quantization for natural images full of colors. However, there is no specific parameter value that is general for all the images. This can be attributed to the fact each image will have specific original color palettes and also that not all objects respond in the same way to color quantization.

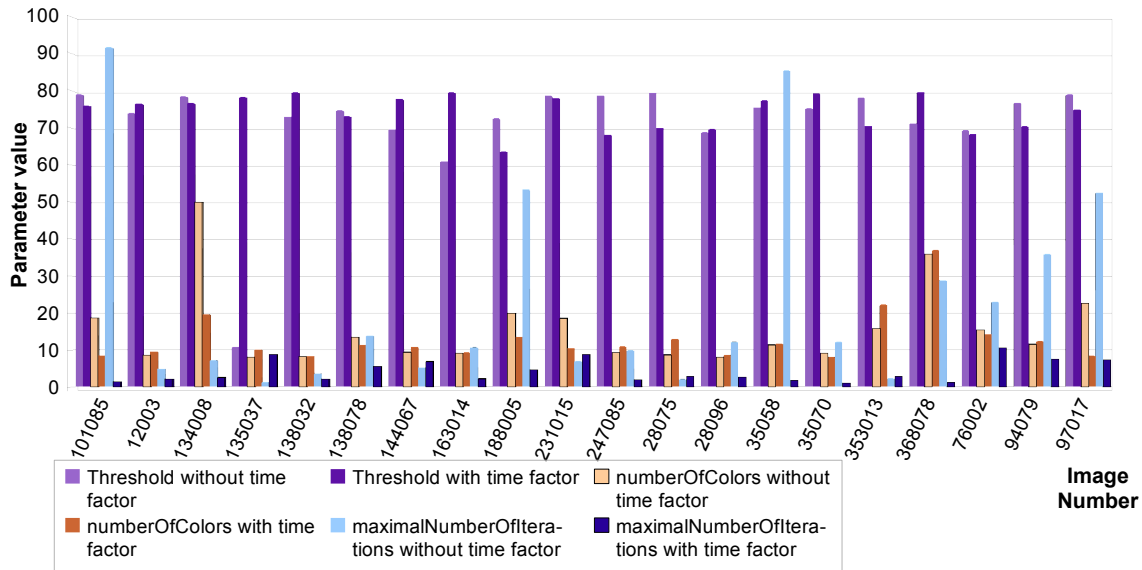


Figure 9: Watershed segmentation evaluation of 20 images with and without time factor showing parameters

The final observation is that the timing factor singles out the iteration parameter for ‘extra’ optimization. The time factor keeps this parameter’s value as low as possible, not more than ten iteration for all the images. Evaluation without the time factor, gives no clear preference for high or low iteration values, which make one conclude that this parameter does not have much importance for segmentation accuracy. Evaluation without the time factor gave no preference for a parameter set with low computation time and as such did not consider low iteration values as an important target for optimization. Each image took about 20 minutes to evaluate when a time factor was not used and the evaluation time reduced to approximately half this value when this factor was included. Evaluation time was consistent with or without the time factor’s inclusion.

The k -means algorithm, employed as an initial smoothing/color quantization process in the Watershed algorithm, can also be augmented by another smoothing stage that uses a k -nearest-neighbor algorithm to smooth out the image further. Again, the time factor was included in GA optimization. It was found that the threshold delta parameter had little effect on the results. This parameter stops the processing iterations if the palette changes between the iteration is less than this value and in the implementation it was set to a range between 0.1 and 1.0. Again other parameters apart from the number of iterations were unaffected by the inclusion of the time factor. However, the evaluation time now decreased dramatically, as Fig. 10 illustrates.

3.2 Discussion

Including a time factor into the GA cost function has three potential advantages: 1) It can increase the convergence speed because less important parameters are not explored in such detail; 2) It arrives at computationally efficient parameter sets; and 3) If GA parameter settings with and without the time factor are examined, insight can be had into insignificant parameters. Graph-based segmentation with a time factor did not yield any noticeable difference in parameter settings from not using a time factor, as there is no parameter that strongly influences the evaluation times. However, in other algorithms such as mean-shift segmentation all three benefits occur.

4 CONCLUSIONS

To arrive at a best-fit parameter configuration, in the face of a diversity of parameter-dependent results is a time-consuming task, yet seems necessary if quantitative evaluation is to become standard. Apart from the speed up over an exhaustive search, using a genetic algorithm has highlighted the importance of parameter settings and the criticality of a particular parameter over another. However, a genetic algorithm is by no means guaranteed to find an optimal solution and this study experimented with a polishing algorithm to improve the solution. The addition of a time factor into the GA cost function, does not just select for parameter settings that improve the throughput but also rationalizes the selection so that relatively unimportant parameters are not explored in too great detail.

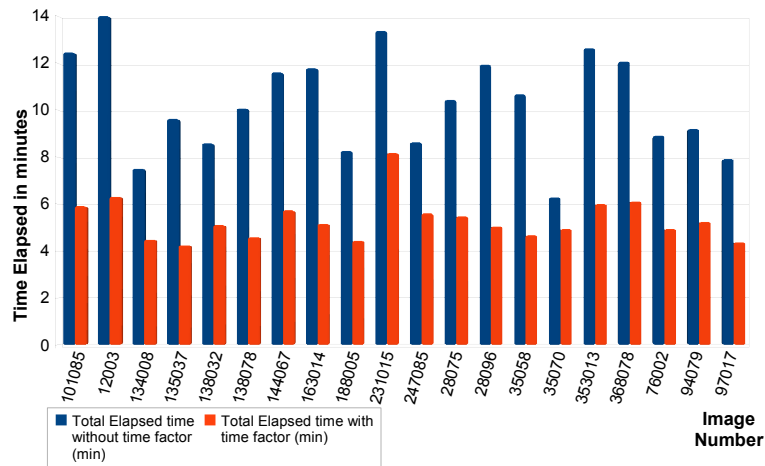


Figure 10: *K*-means segmentation evaluation timings of 20 images with and without time factor optimization

Therefore, we conclude that when batches of images are being processed, including a time factor in the assessment is generally beneficial both to segmentation quality and image processing throughput.

REFERENCES

- Borra, S. and Sarkar, S. (1997). A framework for performance characterization of intermediate-level grouping modules. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(11):1306–1312.
- Chabrier, S., Laurent, H., and Emile, B. (2005). Performance evaluation of image segmentation: application to parameters fitting. In *13th Europ. Sig. Proc. Conf.*
- Comaniciu, D. and Meer, P. (2002). Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(5):603–619.
- Everingham, M., Muller, H., and Thomas, B. (2001). Evaluating image segmentation algorithms using monotonic hulls in fitness/costs space. In *12th British Machine Vision Conf.*, pages 363–372.
- Felzenszwalb, P. F. and Huttenlocher, D. (2004). Efficient graph-based image segmentation. *Int. Journal of Computer Vision*, 29(2):167–181.
- Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley.
- Guyon, I., Markhou, J., Schwartz, R., and Vapnik, V. (1998). What size test set gives good error rate estimates? *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(1):52–64.
- Konishi, S., Yuille, A. L., Coughlan, J., and Zhu, S. C. (1995). Fundamental bounds on edge detection: An information theoretic evaluation of different edge cues. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 573–579.
- Levachkine, S. and Sossa, J. (2000). Image segmentation as an optimization problem. *Computation and Systems*, 3(4):245–263.
- Martin, D., Fowles, C., Tal, D., and Malik, J. (2001). A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Int. Conf. on Computer Vision*, pages 416–423.
- Mühlenbein, H. and Schlierkamp-Voosen, D. (1993). Predictive models for the breeder genetic algorithm: I. continuous parameter optimization. *Evolutionary Computation*, 1:25–49.
- Nelder, J. A. and Mead, R. (1965). A simplex method for function minimization. *Computer Journal*, 7:308–313.
- Olague, G. (2007). Evolutionary computer vision. In *GECCO conf. companion on genetic and evolutionary computation*, pages 3458–3507.
- Polheim, H. (2005). *GEATbxSurvey: Evolutionary Algorithm Toolbox for MATLAB, version 3.7*. Online at <http://www.geatbx.com/docu/index.html>.
- Rees, G. and Greenway, P. (1999). Metrics for image segmentation. In *ICVS Performance Characterization Workshop*, pages 199–210.
- Schoenemann, T. and Cremers, D. (2007). Introducing curvature into globally optimal image segmentation: Minimum ratio cycles on product graphs. In *IEEE 11th Int. Conf. on Computer Vision*, pages 1–6.
- Vincent, L. and Soille, P. (1991). Watersheds in digital spaces: An efficient algorithm based in immersion simulations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(6):583–598.
- Zhang, H., Fritts, J. E., and Goldman, S. A. (2005). A co-evaluation framework for improving segmentation evaluation. In *SPIE Conf.*, volume 5809, pages 420–430.