VCIP 2017

Parameter-free Hierarchical Image Segmentation

S M Abdullah[1], Peter Tischer[1], Sudanthi Wijewickrema[2], Andrew Paplinski[1]

[1] *Faculty of Information Technology, Monash University, Australia*
[2] *Department of Surgery (Otolaryngology), The University of Melbourne, Australia*

[1] {sm.abdullah, peter.tischer, andrew.paplinski}@monash.edu
[2] sudanthi.wijewickrema@unimelb.edu.au

***Abstract*—Images typically have many levels of detail and the suitability of a segmentation depends on application requirements. Thus, it is imperative that the user/application be given the option to select the 'optimal' segmentation that captures the desired level of detail from a set of segmentations. This paper presents a hierarchical image segmentation algorithm that offers this option using the concept of minimum spanning trees. It converts an input image into a tree structure from which a hierarchy of segmentations is obtained through a process of merging. No parameters are used in this process and thus the proposed algorithm can be used on any segmentation dataset as is. The levels are calculated in one pass of the minimum spanning tree and as such, no iterative merging is required. This provides the user with a quick way of segment visualisation. Evaluation results on two popular segmentation datasets show that the algorithm provides competitive results in comparison to other segmentation algorithms.**

***Index Terms*—segmentation, segment visualiser, hierarchical, MST, parameter-free**

I. Introduction

Image segmentation plays a vital role in many computer vision applications (e.g., object detection, object recognition). It attempts to capture important visual perceptual information in an image by representing it using segments that have similar visual properties. Depending on the number of resulting segmentations, image segmentation algorithms can be categorised into two groups: flat-structure algorithms that provide a single segmentation for an image and hierarchical methods that provide a set of segmentations at different levels of detail.

In flat-structure segmentations, typically, the 'best' segmentation for an image is obtained by tuning one or more parameters. For example, Felzenszwalb and Huttenlocher [1] proposed a greedy graph-based algorithm that used an 'observation scale' to determine whether or not to merge two segments. This parameter is dependent on the size and content of an image. Li et al. [2] proposed a graph-based solution based on affinity graph creation. Their approach considered regional information when forming a graph node. To capture this information, instead of considering pixels as nodes in a graph, they used superpixels generated by the SLIC [3] algorithm. To control segment formation, this algorithm used multiple parameters.

Image segmentation however, is an ill-posed problem as there is no unique segmentation of an image. The 'optimal' level of segmentation depends on application requirements and image properties. As such, we believe a single level of segmentation is not adequate to capture perceptually important information from an image. A better option would be allow the user to select the best-suited segmentation according to their requirements from a hierarchy of segmentations at different levels of detail. Typically, in hierarchical algorithms, the first level of segmentation considers each pixel to be a 'segment' in its own right. Then, the segments are merged at different levels, until at the final level, the image itself forms one segment.

Guimarães et al. [4] proposed a hierarchical segmentation algorithm based on the work of Felzenszwalb and Huttenlocher [1]. Instead of using one observation scale as used in the base algorithm [1], they calculated the scale locally based on the internal and external distances between two segments to provide a set of segmentations.

Abdullah et al. [5] proposed a segmentation method based on the concept of mutual nearest neighbours. It chained the image pixels to form segments based on the choice of the number of neighbours to consider. This algorithm is iterative as the output of one level is used to generate segments for the next level.

Shi and Malik [6] proposed a graph cut algorithm known as *Ncut* for image segmentation. This method solved some known issues of early graph cut algorithms, such as partitioning small isolated nodes in the graph. To find the optimal partition of the graph, it solved the Ncut property through a series of linear algebraic equations. This method resembles eigenvalues-based approaches [7] proposed earlier in the literature. One of the drawbacks of the Ncut approach is that, in general, it requires high computational complexity.

In addition to the above-discussed methods, there exist many other well-known methods (Arbelaez et al. [8], [9], and Cousty et al. [10] to name a few), the details of which will not be discussed here due to space limitations.

This paper introduces a hierarchical segmentation algorithm that overcomes limitations of existing methods. The contributions of this paper are as follows. First, our algorithm does not use any parameters and hence it can be applicable to any segmentation dataset as is. Second, it generates all the segmentation levels at once, resulting in speeds comparable to that of flat-structure algorithms. Third, it considers the difference in segment levels as well as the distance between segments when merging, to achieve greater homogeneity in the resulting segments. Fourth, it is independent of the distance function used to measure the dissimilarity among image pixels and hence it is convertible to a clustering algorithm. Last but not least, it includes a segment visualiser which helps the user in selecting the appropriate level of segmentation.

The rest of the paper is organised as follows. The proposed method is described in Section II followed by Section III which discusses the experimental setup, result analysis, and comparisons. Section IV concludes the paper.

II. Proposed Method

The proposed method starts by converting the input image I into an undirected graph $G(V, E)$, where V represents vertices (pixels) of I and E represents the cost to connect two vertices from V. An eight-connected neighbourhood is considered when constructing the graph. Pixel colours are converted to the CIELAB colour space since the RGB colour space is perceptually non-uniform and for natural images, RGB colour components have high correlation [11]. Then, the perceptual colour distance function CIEDE2000 [12] is used to determine edge costs based on the pixel to pixel distance.

The graph is then used to construct a minimum spanning tree (MST). From a graph with M vertices, an MST with $(M-1)$ edges is constructed. The edges of the MST are then sorted in ascending order of the edge cost. The benefits of using MST are twofold. First, it makes the number of edges linear to the number of vertices. Second, it enables us to merge each segment with its spatially-connected nearest neighbouring segment.

The individual vertices are considered as singleton segments (one-element segments) and stored in the vertex list (V_list). In addition to the elements of each segment, this list stores three other important information (index, level, and merging cost) for each segment to facilitate the segment merging process. Each singleton segment is assigned an index to identify itself in the merging process. The level of each segment is set to zero at the start as all singleton segments are on the same level. Merging cost is initially set to zero for all singleton segments. When an edge merges two segments from this list, a new entry is created in the V_list to store the newly formed segment and the indices of the two merging segments are updated to link with the current segment. The merging cost of the new segment is updated to reflect the current edge cost and the new level is decided based on the following rules.

A. *Segment merging at the same level*

When an edge merges two segments which are on the same level, the following rules are observed to determine the new level of the resulting segment.

1) If both segments are on level zero or on the same level, they are merged at the next level.
2) If both segments are in a region of low colour variation they are merged with each other at the current level.

The first rule allows us to increase the level in the segmentation hierarchy. The second rule helps to grow the nearest neighbouring chain of each segment. The current edge cost is compared with the stored merging cost to identify regions of low colour variation.

B. *Segment merging at different levels*

If an edge connects two segments which are on different levels, the maximum level of these two merging segments is the new level at which they are merged and the lower level segment does not participate in any intermediate level mergings.

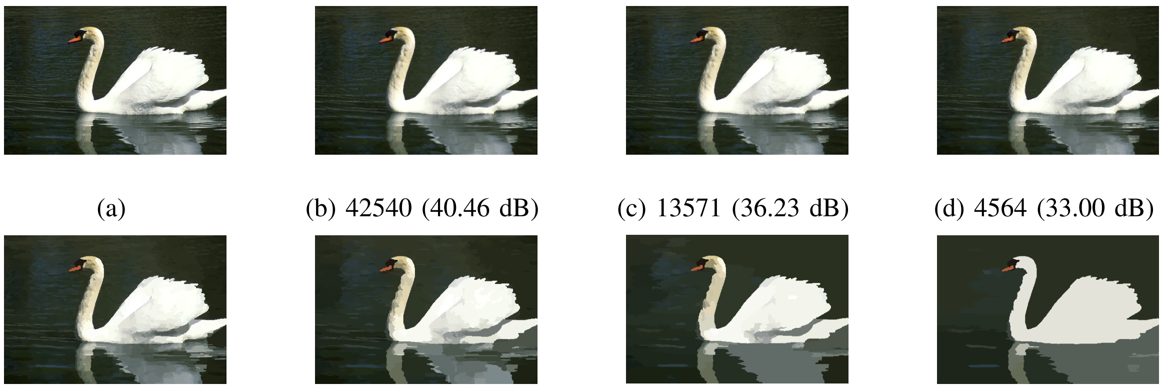
For instance, if a segment S_A is on level 2 and another segment S_B is on level 5, S_A merges with S_B at level 5 and S_A does not participate in any merging at levels 3 and 4. Instead, it copies itself in these two levels. Here, the level difference is treated as a cost in the merging process along with the edge cost. This merging technique may create many singleton/small segments in the segmentation hierarchy. The number of singleton segments at higher levels is a good indicator of whether impulse noise is present in the image.

The union-find algorithm [13] was used on the V_list to efficiently find merging segments.

C. *Post-processing: merging small segments*

Some unmerged segments, mostly on the object boundary, may remain until the final level. This occurs mostly on natural images where object boundary pixels have higher variations than other neighbouring pixels or pixels that belong to more than one segment. To merge these unmerged segments, we apply a heuristic post-processing approach. As the final level (L) of the hierarchy merges all segments into one, this post-processing approach is applied only on levels $L-1$ to 1. It searches for an edge which connects two segments that are on the same level (l_i) with at least one of the segments having a size less than 2^{l_i}. Then, it merges these two segments at this level (l_i). The motivation for this is that, at any level (l), a segment should have at least 2^l pixels in it to reduce the number of segments by half at each consecutive level. This post-processing significantly reduces unmerged segments generated at the object boundary, especially at higher levels.

Figure 1 presents all the levels of segmentations generated by our method except the final level for a representative image from the Berkeley segmentation dataset and benchmark [14]. Each pixel has been replaced by the mean colour of the pixels in that segment. Note that at each level, the number of segments is reduced by more than half. The numbers inside the brackets represent the peak-signal-to-noise ratio (PSNR) values. At any level, the PSNR value is computed using the colour values of generated segments and the actual pixel values. The PSNR measures how well the mean pixel value in the segment approximates the pixel values in the segment. This figure also shows how our method provides a quick segment visualiser from which a user can select the best segmentation level depending on application requirements.



(a) (b) 42540 (40.46 dB) (c) 13571 (36.23 dB) (d) 4564 (33.00 dB)

(e) 1591 (29.75 dB) (f) 497 (25.63 dB) (g) 136 (23.53 dB) (h) 24 (20.01 dB)

Fig. 1. (a) an image from the Berkeley segmentation dataset. (b)-(h) represent each level of segmentation by the proposed method excluding the final level. The number of segments is given below each image along with the PSNR value in brackets. Segments are shown using the mean colour of the pixels in that segment. Best viewed in colour.

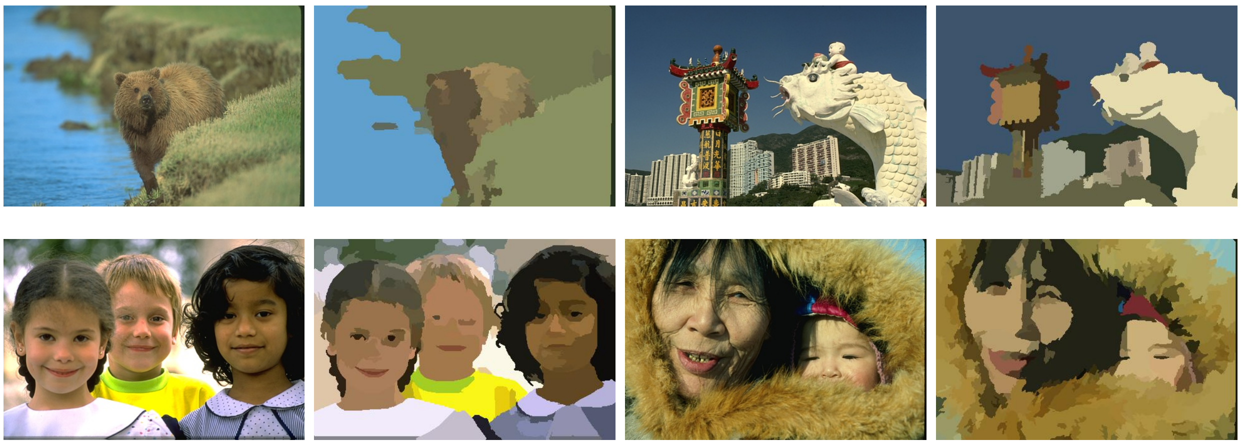


Fig. 2. Visual results of the proposed algorithm on images from the Berkeley BSDS500 segmentation dataset. The first and third columns represent original images from the Berkeley dataset. The second and fourth columns present segmentations of the images in the first and third colums respectively. Each segment is presented by its mean colour. Best viewed in colour.

III. Experimental Results and Comparisons

The proposed method was implemented using the MATLAB software package [15] in a standard desktop environment. Figure 2 presents some visual results from the Berkeley benchmark dataset [14]. The level of segmentation that is shown provides the best visual result. This is a subjective trade-off between the number of segments and generated picture quality.

A. Datasets

We used two different datasets to evaluate the performance of the proposed method. The first dataset is the Berkeley segmentation dataset and benchmark (BSDS500) [14], which contains 200 natural images. This dataset provides various performance metrics such as segmentation cover (Seg_cover) [16] and probabilistic random index (PRI) [17] for evaluating the performance of a segmentation algorithm. The second dataset is the one proposed by Alpert et al. in [18]. This dataset is divided into two parts as a single-object dataset and two-object dataset with each having 100 test images. This dataset uses the average f-measure [19] to report the performance of segmentation algorithms.

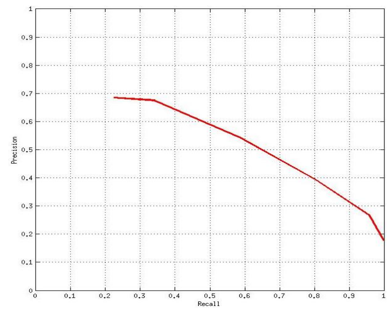


Fig. 3. Precision-recall curve for the BSDS500 dataset

Figure 3 presents the precision-recall curve of the proposed algorithm on the Berkeley dataset. For each test image, the precision and recall value was computed based on the boundary matching between ground-truth segmentations and machine generated segmentations. Then the f-measure, or the

the harmonic mean of the precision and recall values, was calculated.

B. Comparisons

Table I presents the segment region accuracy of several segmentation algorithms when tested on the Berkeley dataset. The proposed method is the only algorithm which is parameter-free and non-iterative (i.e., all levels are generated in a single pass). All the other methods are iterative (i.e., use one level's output to generate the next level). Our method provides the best overall results.

TABLE I
SEGMENT REGION ACCURACY COMPARISON RESULTS ON THE BSDS500 DATASET WITH EXISTING HIERARCHICAL ALGORITHMS

Method	Seg_cover	PRI	Parameter-free	Iterative
Proposed method	0.48	0.80	yes	no
Guimarães et al. [4]	0.42	0.75	no	yes
Ncut [6]	0.45	0.78	no	yes
Abdullah et al. [5]	0.50	0.79	no	yes
α-tree [20]	0.44	0.78	no	yes

Table II shows the comparison results as applied to Alpert's segmentation dataset [18]. For the single-object dataset, our method provides a better score than the mean shift algorithm [21], but not the other two compared methods. For the two-object dataset, a significant improvement is observed where our method provides the best score of all of the compared methods.

TABLE II
SINGLE OBJECT TEST COMPARISON RESULTS ON ALPERT'S DATABASE

	f-measure	
Method	Single-object	Two-object
Proposed method	0.63 ± 0.06	0.82 ± 0.03
Alpert et al. [18]	0.86 ± 0.01	0.68 ± 0.05
Mean shift [21]	0.57 ± 0.02	0.61 ± 0.02
Ncut [6]	0.72 ± 0.02	0.58 ± 0.06

Although the parameters of all other methods need to be tuned, as the proposed method is parameter-free, it can be applied on any segmentation dataset as is. Therefore, it is possible to use our algorithm to quickly explore the natural connectivity among the image pixels at different levels.

IV. CONCLUSION

This paper presented a bottom-up aggregation procedure, in which segments are merged at different levels to form larger segments, exploiting the concept of minimum spanning trees. The algorithm allowed the construction of a segment hierarchy from a single pass of the tree. It is parameter-free, simple, fast and easy to understand. Yet, it outperforms existing segmentation algorithms when tested on two popular segmentation datasets. The proposed method is independent of the cost function and is convertible to a clustering algorithm. It can be generalised to 3D or higher dimensional images. Moreover, in the segment formation process, it is also capable of working with superpixels and vector-valued pixels.

V. ACKNOWLEDGEMENT

This research was funded by an Australian Government Research Training Program Scholarship.

REFERENCES

[1] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[2] A. Li, X. Wang, K. Yan, C. Li, and D. Feng, "Multilevel affinity graph for unsupervised image segmentation," in *Image Processing (ICIP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1264–1268.

[3] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012.

[4] S. J. F. Guimarães, J. Cousty, Y. Kenmochi, and L. Najman, "A hierarchical image segmentation algorithm based on an observation scale," in *Structural, Syntactic, and Statistical Pattern Recognition*. Springer, 2012, pp. 116–125.

[5] S. Abdullah, P. Tischer, S. Wijewickrema, and A. Paplinski, "Hierarchical mutual nearest neighbour image segmentation," in *Digital Image Computing: Techniques and Applications (DICTA), 2016 International Conference on*. IEEE, 2016, pp. 1–8.

[6] J. Shi and J. Malik, "Normalized cuts and image segmentation," *Pattern Analysis and Machine Intelligence, IEEE Trans on*, vol. 22, no. 8, pp. 888–905, 2000.

[7] R. Van Driessche and D. Roose, "An improved spectral bisection algorithm and its application to dynamic load balancing," *Parallel computing*, vol. 21, no. 1, pp. 29–48, 1995.

[8] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 5, pp. 898–916, 2011.

[9] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 328–335.

[10] J. Cousty, L. Najman, and B. Perret, "Constructive links between some morphological hierarchies on edge-weighted graphs," in *International Symposium on Mathematical Morphology and Its Applications to Signal and Image Processing*. Springer, 2013, pp. 86–97.

[11] M. Tkalcic, J. F. Tasic *et al.*, "Colour spaces: perceptual, historical and applicational background," in *Eurocon*, 2003.

[12] G. Sharma, W. Wu, and E. N. Dalal, "The CIEDE2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations," *Color Research & Application*, vol. 30, no. 1, pp. 21–30, 2005.

[13] H. N. Gabow and R. E. Tarjan, "A linear-time algorithm for a special case of disjoint set union," in *Proceedings of the fifteenth annual ACM symposium on Theory of computing*. ACM, 1983, pp. 246–251.

[14] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. 8th Int'l Conf. Computer Vision*, vol. 2, July 2001, pp. 416–423.

[15] MATLAB, *version 8.6 (R2015a)*. Natick, Massachusetts: The MathWorks Inc., 2015.

[16] T. Malisiewicz and A. A. Efros, "Improving spatial support for objects via multiple segmentations," 2007.

[17] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the American Statistical association*, vol. 66, no. 336, pp. 846–850, 1971.

[18] S. Alpert, M. Galun, R. Basri, and A. Brandt, "Image segmentation by probabilistic bottom-up aggregation and cue integration." in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2007.

[19] C. J. V. Rijsbergen, *Information Retrieval*, 2nd ed. Newton, MA, USA: Butterworth-Heinemann, 1979.

[20] P. Soille, "Constrained connectivity for hierarchical image partitioning and simplification," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 7, pp. 1132–1145, 2008.

[21] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 5, pp. 603–619, 2002.