A High Level Merging Tool in Image Segmentation Applications

Philippe Charnier, Ehoud Ahronovitz, Christophe Fiorio

MVA'94 IAPR Workshop on Machine Vision Applications Dec. 13-15, 1994, Kawasaki

A HIGH LEVEL MERGING TOOL IN IMAGE SEGMENTATION APPLICATIONS

Philippe CHARNIER, Ehoud AHRONOVITZ and Christophe FIORIO

LIRMM, Département d'Informatique Fondamentale
UMR 9928 Université Montpellier II/CNRS
161, rue Ada 34392 Montpellier Cedex 5
FRANCE
e-mail: {charnier,aro,fiorio}@lirmm.fr

ABSTRACT

This paper deals with "on the fly" segmentation on sliding images. We describe a quality control industrial application for which that type of image analysis is an elegant solution. Local and global merging techniques on the region adjacency graph are discussed. Only local merging allows us to process an uncompletely known graph. Global merging on the entire graph and "on the fly" local merging give similar segmentation results, and yet the latter is faster.

Keywords: *image analysis, region adjacency graph, "on the fly" processing, quality control.*

INTRODUCTION

An important problem in image analysis is to capture the essential features of a scene. One of the most significant features is the *region* which denotes a homogeneous set of connected points of the image. Usually, we proceed by successive merge operations of regions in order to obtain a partition of the image. Regions growing techniques were first described in [MA68]. These operations are realized in the Region Adjacency Graph [Ros74, Pav77] by merging vertices. There are two essential techniques:

- local merging: we choose randomly a vertex; among all it's adjacent neighbours, if some can be merged, then we select the best if any [Cha94]. Knowing the choosen vertex and its neighbourhood is sufficient in order to start the merging process,
- global merging: we take into account all the pairs of adjacent vertices in the whole graph; if some of these pairs can be merged then we select the best if any [PCCB91]. The entire graph is required when the merging process starts.

In the following, we present these two techniques, and we detail some advantages of local merging. We also describe an industrial application in which local merging was used. It allows us to do "on the fly" processing.

MERGING STRATEGY

Local merging: In local merging, we first get a vertex in the R.A.Graph, called the *reference vertex*, and we try to merge it with one of its neighbours according to a decision criterion. We say that a *turn* was made when all the vertices were scanned, i.e. each vertex was in the state of either a reference vertex or a merged one. We stop as soon as a blank turn was made, which means that all the vertices were scanned and no merging action was performed.

For each reference vertex, we look for a *candidate* to merge. A candidate is an adjacent vertex fitting the decision criterion. If more than one neighbour is candidate, the decision criterion chooses one of them. Then any other vertex, except the newly created, becomes the new reference vertex.

As soon as a try succeeds, another turn becomes necessary. Our experiments show that about a dozen of turns are required for the merge process to complete.

Note that any vertex can be many times a reference vertex but only once in one turn. This does not exclude the possibility for a vertex to be a reference one and to be merged later in the same turn (as a neighbour of another reference vertex).

Global merging: The global merging examines all pairs of adjacent vertices (all the edges) in the whole graph. The decision criterion chooses the one pair to be merged. Unlike the previous case, there is no reference vertex.

All the edges are sorted according to the decision criterion. An initial ordered set of edges is thus created. When a merge occurs, all the incident edges to both vertices are deleted. The incident edges to the newly created vertex[1] are inserted in the ordered set.

In all cases, the merging process continues until there is no more vertices to be merged.

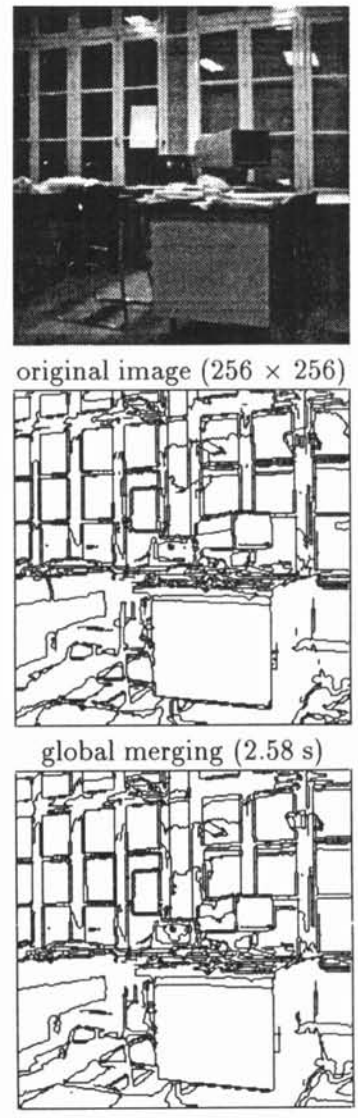
Global versus local: We have studied several decision criteria for both approaches. Globally, they are based on the region's colour or grey level, area, border length, etc.

One can think that with less efficiency in run time, better results will be obtained by the global merging. But our experiments[2] show that the local merging gives as good results as the global one, and still with less run time (see Figure 1).

This is may be due to the difficulty to have a sufficiently precise decision criterion. In fact, we obtain many equivalent results for the best choice, while what we need is a strictly ordered set (a unique best choice). We can say that the more the number of equivalence classes is reduced, the more the results of local merging are close to the global case.

[1]The edges linking the new vertex and all neighbours of the two deleted vertices.

[2]On a Sparc 10 workstation.



original image (256 × 256)

global merging (2.58 s)

local merging (0.29 s)

Figure 1: local vs global merging

Moreover, unlike the global merging, local merging does not need the entire graph: only the reference vertex and its neighbourhood are needed. This can be useful when we need to begin the segmentation process whereas the graph extraction is not fully completed. We just have to wait until the whole neighbourhood is completely known. This can be done easily: a vertex is included in the graph only when the corresponding region is entirely scanned. Thus, all the neighbours, entirely or not yet entirely scanned are known. When all these neighbours will be included in the graph, the vertex will be ready to become

a reference vertex.

APPLICATION

Description: The industrial application that we describe consists in a "on the fly" quality control of a cotton veil. The defects called *seed-coat fragments* in cotton [BG88] determine the quality and the exchange rate of the product. For that, statistical data such as the number of defects, their area, their distribution, and their shape are necessary. Researchers and industrials are highly interested in improving this quality by reducing the seed-coat fragments. Such improvements can be integrated in genetic methods. This detection of defects helps in rejecting low quality cotton, at the beginning of the production process. Up to now, this control was made only after the spinning, which is an expensive step.

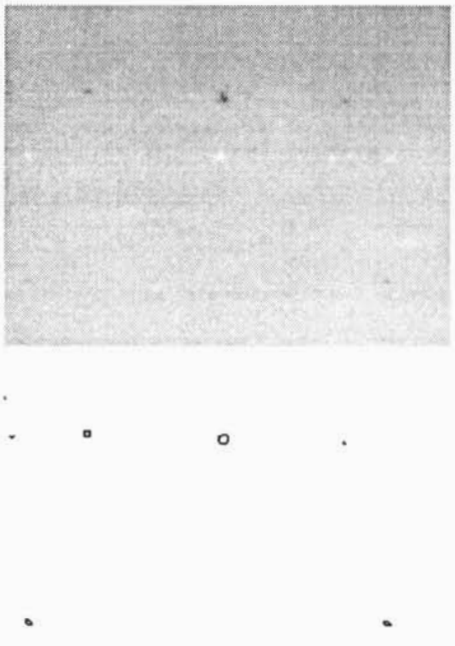


Figure 2: A cotton veil with seed-coat fragments.

Notes on implementation: We have first tested a variety of methods based on local merging. Tests were made on different types of "well-known" images found almost everywhere on the internet.

The R.A.Graph construction goes on along with the image scan, and merging is performed as soon as possible.

We show that the local merging methods that we propose are efficient and are compatible with "on the fly" processing. The R.A.Graph construction algorithm (a full linear method) with the above processing properties, is described in [Cha94].

It is based on an extraction algorithm which scans the image with a two successive lines buffer. The regions' coding is made using interpixel links [Dan82, ABH86] (a contour description in 4 directions evolving in-between the pixels).

Regions are constructed within the scan, then merged at the segmentation step. The R.A.Graph evolves with the closing of regions.

In the seed-coat case, the vertex characteristics allow to find the seed-coat fragments and to distinguish them from the cotton veil. Note that regions sizing wide over the seed-coat fragments area are marked and the associated data structures discarded: they become "black holes" gathering all adjacent cotton regions. Thus, the graph is continuously growing and shrinking:

- growing when regions are constructed by the extraction algorithm,
- shrinking when "deleting" non seed-coat regions. This is done by the segmentation algorithm.

These two steps can be performed as two concurrent processes, if the merging process works on a snapshot of the growing graph. As suggested before, only vertices with known neighbourhood are included in the snapshot. Vertices with incomplete neighbourhood will not be updated by the two processes at the same time. So, the memory used by the application can be bounded if a balance between growing and shrinking is found. This is possible because shrinking needs less run time than growing.

Finally, the life-time of seed-coat regions is user-dependant.

CONCLUSION

The application described above involves many linear algorithms especially when extracting the connected components, creating and updating the graph; linear algorithms dealing with graphs are essential to improve efficiency.

Local merging with "on the fly" processing can operate on images with undefined limits such as in the cotton case (see Figure 2). Regions are constructed and discarded according to selection criteria such as: keep those regions with dark grey colour, limited area or other specific characteristics depending on the application.

The tests of different merging procedures, all lying upon a unique kernel, show that adaptibility is a keyword in our method. Any region growing segmentation algorithm can be used as long as it operates on the R.A.Graph and/or the basic characteristics attached to the connected components detected with the R.A.Graph.

Moreover, it is the first time, to our knowledge that the association between merging tools, the R.A.Graph and sliding images is made.

ACKNOWLEDGMENT

E. HEQUET (CIRAD-Montpellier) suggested this application.

REFERENCES

[ABH86] E. Ahronovitz, M. Bertier, and M. Habib. Contour coding for image manipulation and compression. *IEEE IAPR*, 2(742), 1986.

[BG88] J. D. Bargeron and T. H. Garner. The role of seed-coat and mote-fragment neps in yarn and fabric imperfections: a survey. In *Beltwide Cotton Production Research Conferences*, pages 586–591. Memphis, Tn (USA), National Cotton Council, 1988.

[Cha94] P. Charnier. *Outils algorithmiques pour le codage interpixel et ses applications.* PhD thesis, Université Montpellier II, 1994.

[Dan82] P. Danielsson. An improved segmentation and coding algorithm for binary and nonbinary images. *IBM Journal of Research and Development*, 26(6), 1982.

[MA68] J. L. Muerle and D. C. Allen. Experimental evaluation of techniques for automatic segmentation of objects in a complex scene. In G. C. Cheng et al, editor, *Pictoral pattern recognition*, pages 3–13, 1968.

[Pav77] T. Pavlidis. *Structural Pattern Recognition.* springer verlag, 1977.

[PCCB91] M. Popovic, F. Chantemargue, R. Canals, and P. Bonton. *Several approaches to implement the merging step of the split & merge region segmentation*, pages 399–412. Elsevier Science Publishers B.V., 1991. ©Eurographics Association.

[Ros74] A. Rosenfeld. Adjacency in digital pictures. *Inform. and Control*, 26:24–33, 1974.