A NEW CONTOUR SIMPLIFICATION FILTER FOR REGION-BASED CODING

V. A. Christopoulos[1], C. A. Christopoulos[2], J. Cornelis[1], A. N. Skodras[3]
[1]Vrije Universiteit Brussel, VUB-ETRO (IRIS), Pleinlaan 2, 1050 Brussels, Belgium
[2]Ericsson Telecom AB, HF/ETX/MN, S-126 25 Stockholm, Sweden
[3]University of Patras, Electronics Laboratory, Patras 26110, Greece
Tel: +32 2 6292982; fax: +32 2 6292883
e-mail: vschrist@etro.vub.ac.be

ABSTRACT

In region-based (RB) coding, the image is divided into a number of various-shaped regions, which are then treated as objects and coded. Experimental results show that a bottle-neck in RB coding at very low bitrates is the amount of shape information, represented by the contours separating the regions, which has to be coded. This paper presents a new filter for contour simplification. The filter reduces the number of contour points by an average of more than 30%. A simplification example shows that its use does not affect subjective image quality.

1. INTRODUCTION

In region-based (RB) coding, the image is segmented into a number of various-shaped regions with slowly varying image intensity, which are then treated as objects and coded. The contours separating the regions are coded, e.g. by chain codes, while the image intensity inside an image region is approximated by a linear combination of base functions. Experimental results show that, at high compression ratios (CRs) and if no special measures are taken, the segments contour information alone represents more than 60% of the coded data [1]. The rest of the compressed data (region texture, motion, etc.) can usually be compressed very heavily. In order to reduce the number of bits for contour coding, a post-segmentation simplification of the contours is needed, especially for complex images.

In the following section, we introduce a new filter for contour simplification. The important properties of the simplification filter are reviewed and a description of how the filter works is given.

In section 3, some simplification results are given together with a comparison with the simplification filter described in [2] and the thinning algorithm described in [3]. The results show that the proposed filter reduces the number of contour points (CPs) by an average of more than 30% while keeping almost the same visual quality.

2. THE NEW CONTOUR SIMPLIFICATION FILTER

The contour image produced by fast RB segmentation schemes is in general not thin (Fig. 2). Apart from some spark edges, it also contains some thick segments, which correspond to very small regions (one pixel thick). This leads to redundant contour information that has to be coded. A thinning algorithm [3] may be applied to the contour image, in order to solve this problem (Fig. 4). But even after the thinning there will be some spark CPs in the contour image, which have to be removed.

The proposed filter is a combination of contour thinning and contour simplification. This new filter when applied recursively to a contour image produces a simplified thinned contour image. The novel idea is that instead of deciding upon removing or not the CP which has been encountered, it decides upon removing or not any of its 8-connected neighbour CPs. This leads to a speed-up (experimentally it has been found that more than 95% of the finally removed CPs are already removed during the first pass).

2.1 Properties of the filter

The proposed filter has been experimentally proven to possess the following properties:

- it does not rely on any particular segmentation algorithm and it is generic with respect to the input images
- it removes all the spark contour points
- it preserves the geometrical relationship between different regions
- it guarantees that no holes between contours are created, and regions do not split into smaller ones

The last property is very important since creation of holes in between contours, or splitting of regions into smaller ones, could lead to a wrong final segmentation.

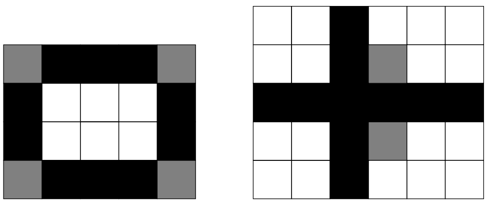
2.2 Description of the filter

The proposed filter works as follows: The binary contour image is scanned from left to right and top to bottom until the first CP is encountered. Let us suppose that a CP (x, y) has been encountered. For each of its 8-connected contour neighbours (x_i, y_i), $1 \le i \le 8$, N_i is calculated, where N_i is the number of 8-connected contour neighbours of CP (x_i, y_i), which do not belong to the neighbourhood of CP (x, y). If N_i is equal to zero, the neighbour point (x_i, y_i) is removed. The four corner CPs of Figure 1.a (coloured in gray) will be removed based on this criterion.

If N_i is not equal to zero but its contour neighbours are indirectly connected with the CP (x, y) via another neighbour (x_j, y_j), $1 \le j \le 8$, $j \ne i$, (this means that the removal of the neighbour point (x_i, y_i) will not cause any discontinuity), the neighbour point (x_i, y_i) is removed. The CPs coloured in gray in Figure 1.b will be removed based on this criterion.

The neighbour (x_i, y_i) will remain in the contour image if none of the above is true.

The procedure is repeated until no CPs are removed. Experimentally it has been found that this normally occurs after less than 10 iterations, depending on the complexity of the contour image.



(a) (b)

Figure 1. Black squares represent CPs. Gray squares represent CPs which have to be removed.

The filter differs from the one proposed [2], since it is applied on the contour (binary) image and not on the label image. Additionally, it converges faster, since multiple CPs are removed each time the filter is applied, as explained above, and it produces a much smaller number of CPs without any distinguishable difference compared to the results of [2].

3. EXPERIMENTAL RESULTS

Figure 2 shows a contour image produced by the segmentation algorithm described in [1]. This image consists of 6790 CPs, which have to be coded. Figure 3 shows the contour image produced after the *majority* filter described in [2] has been applied on the label image. It consists of 6145 CPs (reduction 9%). The filter converges after 8 iterations. Figure 4 shows the same image after the thinning algorithm described in [3] has been applied. It consists of 4571 CPs. This thinned image although consisting of 32% less CPs, still contains some spark CPs which have to be removed. The contour image produced after the application of the proposed filter on the contour image given in Figure 2, is shown in Figure 5. It consists of 4548 CPs (reduction 33%), while the number of iterations needed for convergence is 4. Application of the thinning algorithm described in [3] on the image shown in Figure 5 does not affect the image anymore, proving that the image is adequately thinned. Application of our new filter on the thinned image shown in Figure 4 further reduces the number of CPs by 43.

Figures 6, 7 and 8 show the reconstructed images based on the contour images shown in Figures 2, 3 and 5. Each region has been filled with its mean gray value. Similar reconstructed quality is demonstrated by both the *majority* and our filter. On the other hand it is observable that most of the noticeable artefacts, with respect to the reconstructed image shown in Figure 6, are due to the removal of CPs such as those shown in Figure 1.a. A correction of the simplified image by filling such points before the reconstruction (Figure 9) can remove most of these artefacts (Figure 10). It is also of great importance to mention here that no additional contour information has to be send to the decoder for the correction, since the correction can take place in the decoder side before the reconstruction.

The filter has also been tested with other images than the “cameraman”. Table 1 shows some of the obtained results. For the calculation of the CR, we have assumed that 1.6 bit per contour pixel (bpcp) is needed (although 1.2 bpcp is obtained in [6]) and 8 bits for storing the mean value of each region. A reduction of the amount of the CPs by more than 30% is demonstrated.

The filter has also been tested on images produced by the segmentation algorithms described in [4] and [5]. Similar results have been obtained.

4. CONCLUSIONS

A new filter for simplifying the contours produced by any segmentation algorithm for RB coding schemes is presented. This filter is designed to simplify while thinning the contour (binary) image, without introducing any visible quality distortion. Experimental results have shown that it helps the contour coding part of any RB coding scheme, by reducing the amount of CPs by an average of more than 30%, while keeping almost the same visual quality. The filter outperforms existing contour simplification schemes in speed, number of retained CPs and quality of the simplified image.

ACKNOWLEDGEMENTS

This work was supported by the EC Human Capital and Mobility programme under the contracts ERBCHBGCT930260 and ERBCHRXCT930382 and by the EC ACTS project SCALAR under the contract AC077.

REFERENCES

[1] C. A. Christopoulos, A. N. Skodras, W. Philips, J. Cornelis, A. G. Constantinides, “Progressive Very Low Bit Rate Image Coding”, *Int. Conference on Digital Signal Processing*, June 26-28, 1995, Limassol, Cyprus, pp. 433-438.

[2] C. Gu and M. Kunt, “Contour Simplification by a new nonlinear filter for region-based coding”, *Visual Communications and Image Processing*, VCIP’94, Chicago, U.S.A., September 1994.

[3] P. K. Saha, B. Chanda and D. Dutta Majumber, “A single scan boundary removal thinning algorithm for 2-D binary object”, *Pattern Recognition Letters* 14 (1993), pp. 173-179.

[4] Biggar M. J., Morris O. J., and Constantinides A. G., “Segmented image coding: performance comparison with the discrete cosine transform”, *IEE Proceedings, 135, Part F*, 1988, No. 2, 121-132.

[5] Marques F., Cunillera J. and Gasull A., “Hierarchical segmentation using compound-Markov random fields”, *Proc. of ICASSP 92*, 3, San Fransisco, 53-56.

[6] M. Eden, M. Kocher, “On the performance of a contour coding algorithm in the context of image coding - Part I: contour segment coding”, *Signal Processing*, Vol. 8, No. 4, July 1985, pp. 381-386.

Image	Output of segmentation (method described in [1])			Results after applying the filter described in [2]			Results after applying the proposed filter		
	#Regions	#CPs	CR	#Regions	#CPs	CR	#Regions	#CPs	CR
Cameraman	300	6790	39.5:1	284	6145	43.3:1	295	4548	54.4:1
Miss America	250	6328	43.2:1	239	5673	47.7:1	245	4257	59.8:1
Claire	200	5178	53.0:1	191	4724	57.7:1	196	3560	72.2:1

Table 1. Comparative results between the filter described in [2] and the proposed filter.

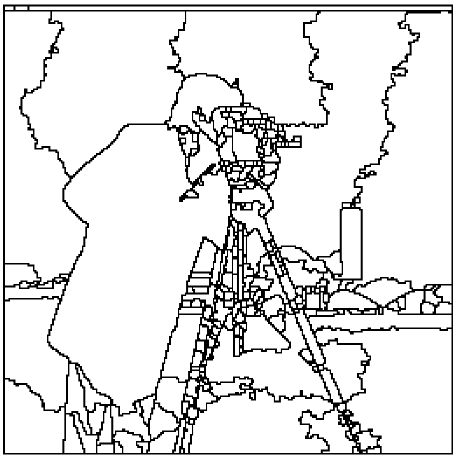


Figure 2. Contour image produced by the algorithm described in [1].

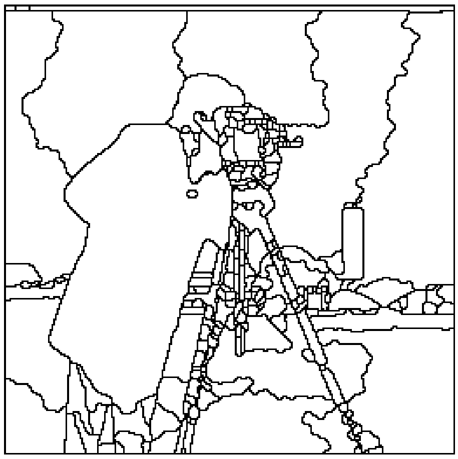


Figure 3. Contour image produced after the filter described in [2] has been applied.

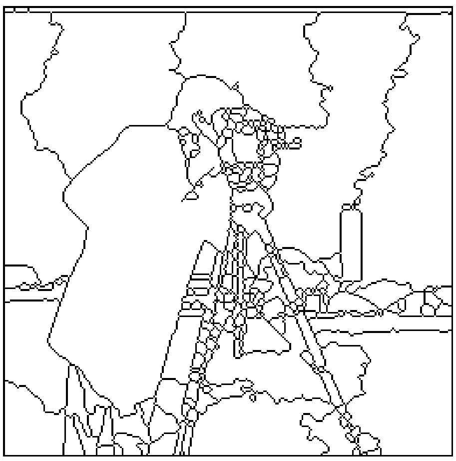


Figure 4. Contour image after thinning of the image shown in Fig. 2 with the algorithm described in [3].

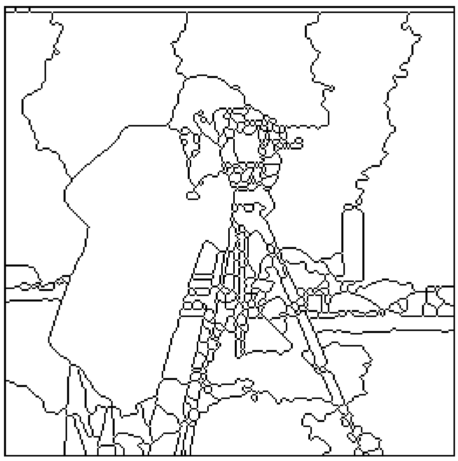


Figure 5. Contour image produced after the proposed filter has been applied on the image shown in Fig. 2.

Figure 6. Reconstructed image based on the contour image shown in Fig. 2.

Figure 7. Reconstructed image based on the contour image shown in Fig. 3.

Figure 8. Reconstructed image based on the contour image shown in Fig. 5.

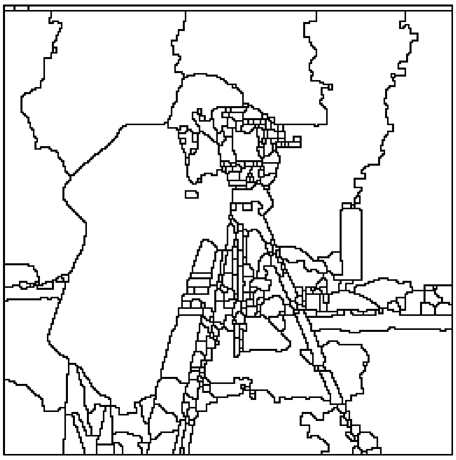


Figure 9. The simplified image shown in Fig. 5 after correction.

Figure 10. Reconstructed image based on the contour image shown in Fig. 9.