Image and Vision Computing 19 (2001) 427–433

www.elsevier.com/locate/imavis

A self stabilizing robust region finder applied to color and optical flow pictures

M. Ben-Ezra[a,*], M. Werman[a], Y. Bar-Yam[b]

[a]School of Computer Science and Engineering, The Hebrew University of Jerusalem, 91904 Jerusalem, Israel
[b]New England Complex Systems Institute, Cambridge, MA,USA

Received 17 September 1997; revised 9 August 2000; accepted 4 September 2000

Abstract

This paper presents a new robust yet very simple algorithm to find a homogeneous region in an image. The algorithm is based on iteratively growing and shrinking a region until a stable state of the algorithm is reached and good fit is found. We have shown that the algorithm works very well in color and optical flow pictures, and overcomes some of the drawbacks of classic seed growing algorithms. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Segmentation; Region growing

1. Introduction

One of the most important preliminary steps leading to the possibility of analyzing images is to group pixels together into regions of similarity, (segmentation). There are many methods of segmentation using domain specific and high level processing, but there are many classes of segmentation problems that can be solved, either completely or as a first step, using lower level processing. There are scores of methods and papers describing threshold, edge and region based methods each of them using different dominant features for segmentation and each having its own set of good points and drawbacks. The most direct methods of segmentation are region growing, region splitting and split and merge methods, for surveys see Refs. [1,2] or for some more recent algorithms [3–5]. Region growing is a low level, bottom up and easily parallelizable method to find meaningful parts of a scene. However starting with a particular seed pixel biases the resulting regions computed, with several undesirable effects, such as; different choices of seeds may give rise to different segmentation results and regions that arise from seeds starting on edges are often mixtures of several different regions. This paper presents a new algorithm that overcomes these problems and is simple, robust and gives very good results.

2. The algorithm

The algorithm that we propose starts a region as a small seed (usually a block of a few pixels) and grows and shrinks the region trying to settle on to a region that fits a specific type of model.

The algorithm Γ iterates the following three stages until it reaches a steady state. R is the region under consideration:

1. *Fit* — fit a model to R.
2. *Shrink* — delete from R pixels that do not fit the model.
3. *Grow* — add to R pixels that neighbor points of R and are close to the model.

The idea behind the algorithm is that the region can both shrink (more dependent on the model than regular split methods) and grow until the algorithm reaches a steady state where the fit is sufficiently good, where the goodness depends on a model that is dynamically changing to reflect its current support. Thus, the starting point is not that important in the final model and it tends to settle in large homogeneous regions. (The reason that we do not start with a very large support in the beginning is that this causes the algorithm to miss some of the regions.) The dynamics of the algorithm let it recover from bad choices so that even when the region extends over too many different parts and noise it will usually recover by deleting all except the

* Corresponding author.

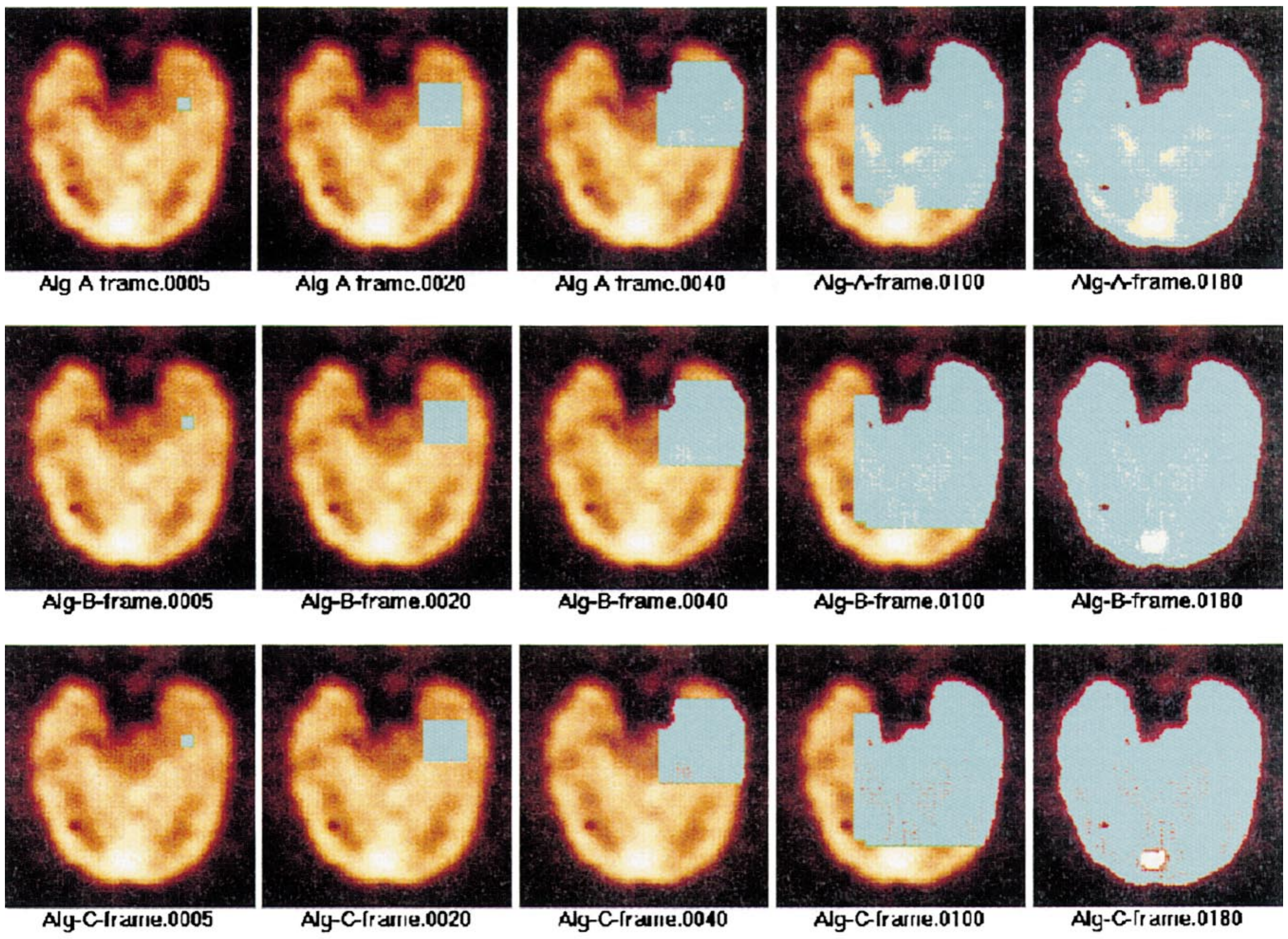

Fig. 1.

major homogeneous part. We have shown how this algorithm was implemented in order to find significant regions in both color and optical flow images.

3. Color

In order to use Γ in color images we use the $L^*a^*b^*$ color model, (the Euclidean distance using the non-linear $L^*a^*b^*$ model is closer to human perception than the linear models such as RGB). The conversion algorithm from RBG to LAB can be found on-line at [6]. The distance to the model used is a weighted sum of the squared differences of the *ab* fields and the luminance field (L) so that when the color is close to gray only L is used and when it is full of color only the *ab* fields are used. The model used is a constant so that a homogeneous region should be a single (up to intensity) color.

4. Optical flow

In order to use Γ in optical flow images (dx, dy) we use the epipolar constraint as the model, so that a region should come from a rigid object. The distance to the model used is the squared value of the value $pF(p + (\mathrm{d}x, \mathrm{d}y))$ where F is the computed normalized fundamental matrix, (0 when p and $p + (\mathrm{d}x, \mathrm{d}y)$ satisfy the epipolar constraint). The epipolar constraint which is expressed by the fundamental matrix simply says that a stationary point in the world and its projection into the image plane in two different time instances forms a plane (true for every three points). This plane intersects the image planes and forms a line on each plane, these lines are named epipolar lines. (The fundamental matrix maps points in one image into lines on the second image. The fundamental line constraint expresses the fact that a dot-product of a point and a line is zero if the point is located on that line).

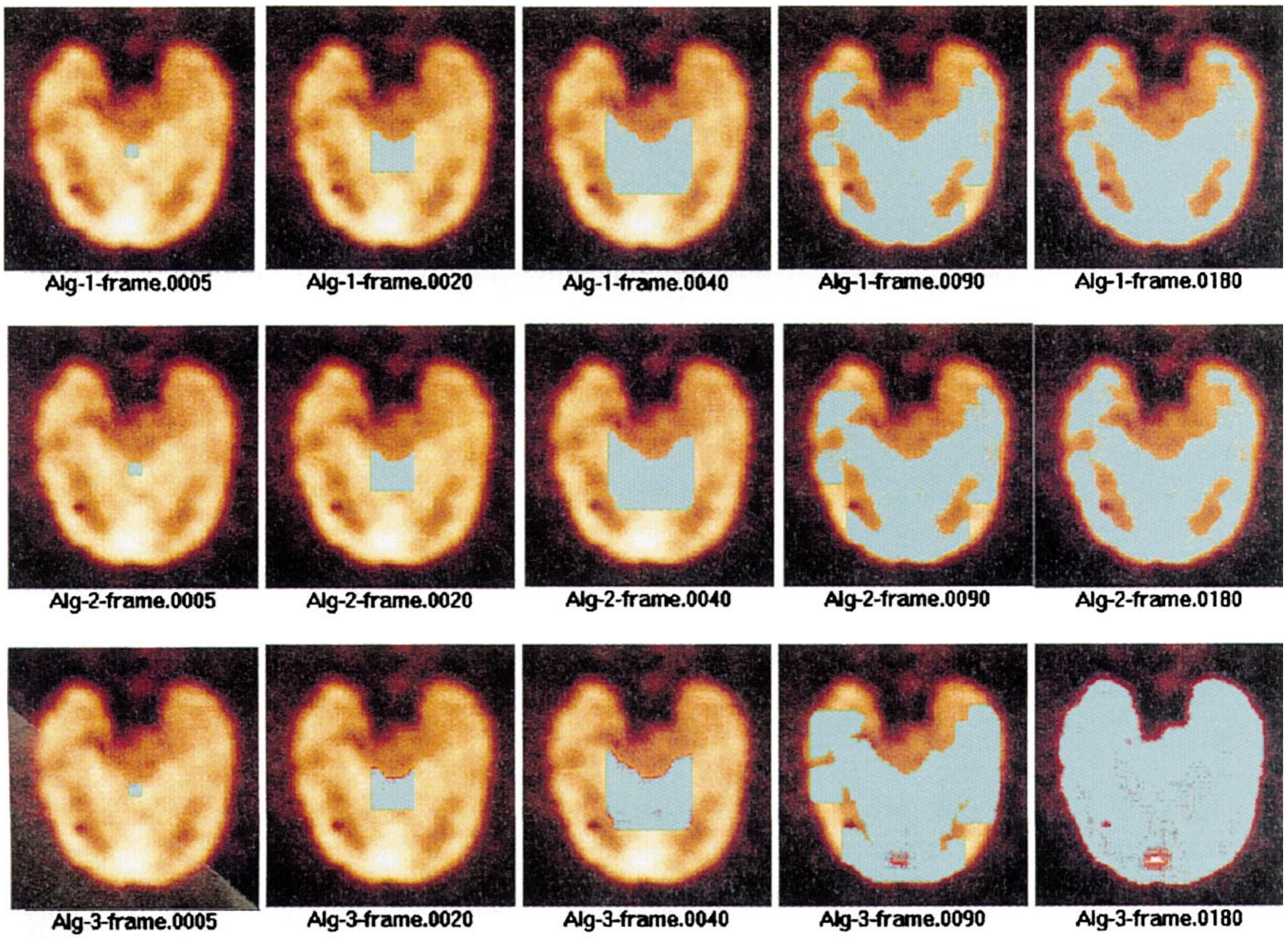

Fig. 2.

5. Implementation details

The model is computed using all the pixels in the region of support using a least squares algorithm. All the information needed to compute the model parameters are the moments and these are updated with the insertion and deletion of each of the pixels into the region, making the re-calculation fast and easy.

6. Results

We show results of using this algorithm on color and optical flow images. The best way to see the dynamics of the algorithm is to look at the MPEG sequence of the iterations of the algorithm, we present here some of elements of these sequences. The optical flow was computed using an algorithm based on correlation using a pyramid citeel, e2.

6.1. False color segmentation — brain PET image

We used the PET image to show the robustness of the Γ algorithm. Algorithm Γ is compared to other common region growing algorithms, notice its stability, the starting point does not make much of a difference and its self stabilizing behavior. The PET image results are shown in Figs. 1–3. Each figure shows the evolution of the segmented region in three different algorithm, the last image in each sequence is in its steady state:

Algorithm-1 is classic region growing where all neighboring pixels that are within a given distance from the seed pixel(s) are added.

Algorithm-2 is classic region growing with model re-calculation at each iteration. The algorithm adds all neighboring pixels that are up to a given distance from the current average.

Algorithm-3 is the Γ algorithm. The adding criteria is the same as Algorithm-2, however the ability of the new algorithm to delete pixels from the region leads to its robustness compared to the previous algorithms.

Cyan denotes the current region, green denotes pixels that just entered into the region and red denotes pixels that were just deleted from the region.

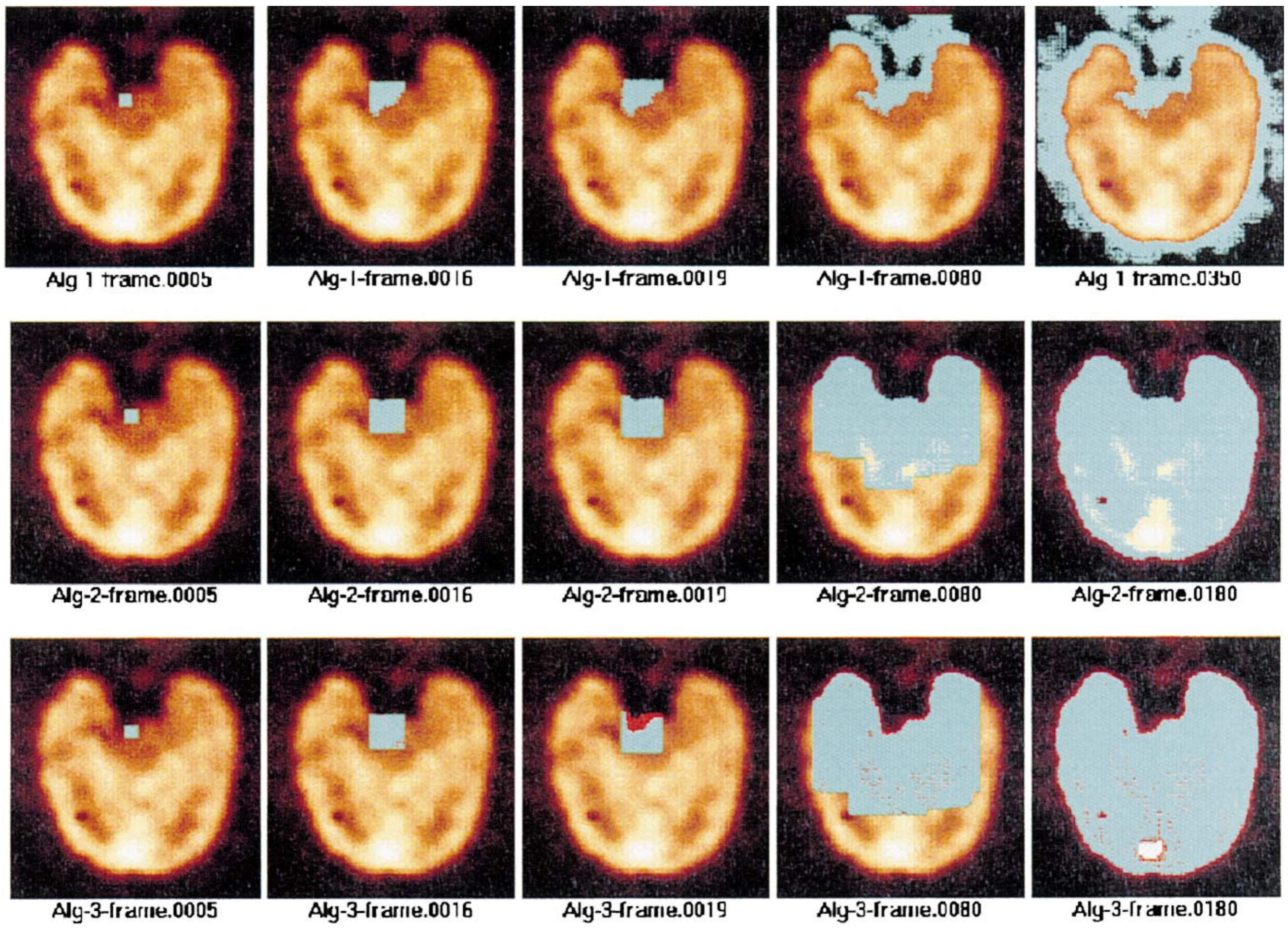

Fig. 3.

Fig. 1 starts with a 'good' seed (found by trial and error), we see that all three algorithms captured approximately the same region. Fig. 2 starts approximately at the middle of the image, we see that the seed bias leads to a great difference in the result in the first two algorithms while the Γ algorithm performs very well.

Fig. 3 starts with a 'bad' seed — dark and close to the edge. We see that Algorithm-1 completely failed. Algorithm-2 could not completely recover from the seed error and grouped non-relevant pixels while missing others, while the Γ algorithm manages to recover from the initial error (frame 19) and find the desired region.

For applications such as 3D modeling from PET/MRI slice images this robustness is essential. (All figures and all algorithms used exactly the same distance metric and parameters).

6.2. Color segmentation — flowers

Fig. 4 shows the same three algorithms applied to a colorful flower image. A 'bad' seed was chosen and the final segment was cut out of the image. We see that Algorithm Γ was able to recover from the initial error while the first algorithm failed and the second algorithm includes different colors.

6.3. Optical-flow: Lucy

Fig. 5 shows the same three algorithms applied to an optical flow image that was derived from two frames from the Lucy sequence [7,8]. The part not in the region is masked out of the picture.

When the region support became large enough to distinguish between Lucy's motion and the room's motion it already included a large part of Lucy's motion. We see that Algorithm-1 was unable to recover from the initial error. Algorithm-2 partially recovered and Algorithm Γ was able to fully recover. Note that Lucy's pivot foot was considered part of the background as it had no movement except for the camera motion.

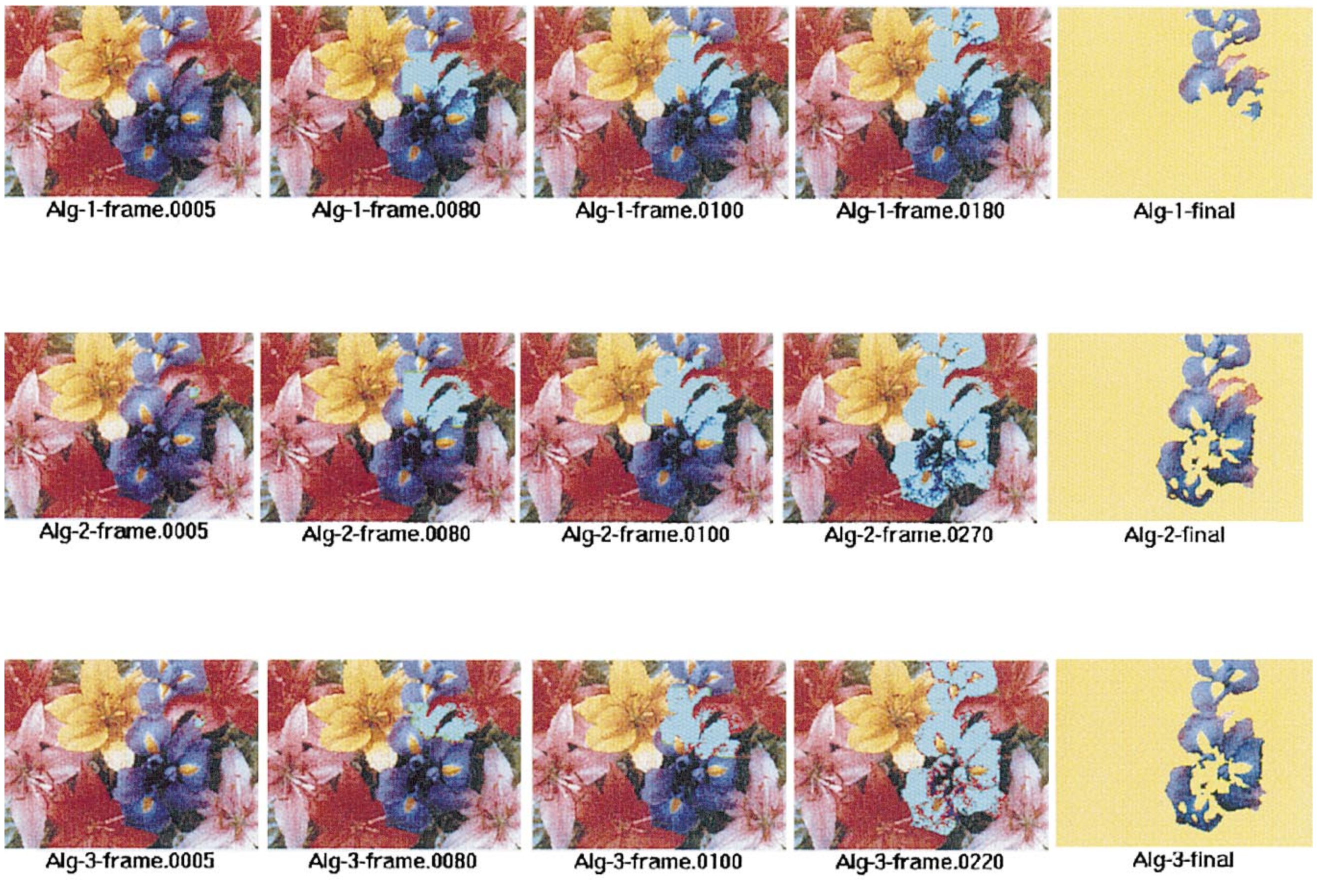

Fig. 4.

In Fig. 6 we see:

1. The two frames that were used to create the optical flow.
2. The generated optical flow represented as an arrow image.
3. The optical flow coded as a color image: the hue denotes the direction of the arrow, and the intensity denotes the length of the arrow — this presentation gives a pixel-dense map of the optical flow and it is often easy to see the irregularities in the hue.
4. The region that was found by the algorithm.
5. The region that was rejected.

6.4. Optical-flow: Car

Fig. 7 shows additional results of the Γ algorithm applied to an optical flow of a 3D scene with a complex motion. Camera motion includes translation, rotation and scale. Car motion was translation. Note in the previous scene, Lucy was not a rigid body, but this time both the background and the Car are rigid.

In Fig. 7 we see:

1. The two frames that were used to create the optical flow.
2. The generated optical flow represented as an arrow image.
3. The optical flow coded as a color image.
4. The region that was found by the algorithm.
5. The region that was selected when the seed was located within the moving car area.

7. Conclusions

We described a simple though robust algorithm for region finding and applied it in different domains — pseudo color, color and optical flow images. We have compared the performance of this algorithm to other simple region growth algorithms and demonstrated the importance of its self stabilizing property to the final result.

The simplicity of the algorithm makes it a natural choice for low-level real-time segmentation on high speed architectures such as cellular automata and DSP's.

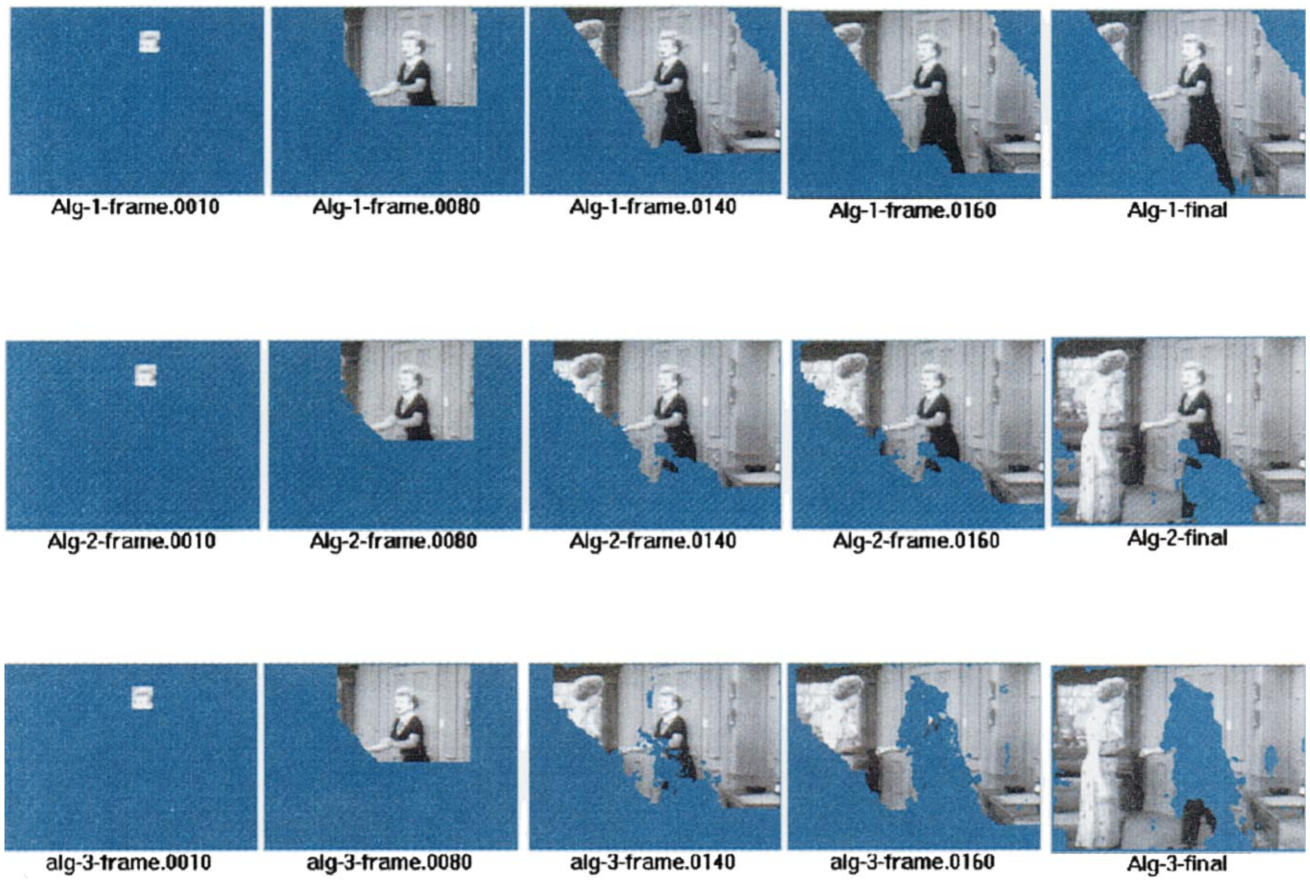
Alg-1-frame.0010
Alg-1-frame.0080
Alg-1-frame.0140
Alg-1-frame.0160
Alg-1-final
Alg-2-frame.0010
Alg-2-frame.0080
Alg-2-frame.0140
Alg-2-frame.0160
Alg-2-final
alg-3-frame.0010
alg-3-frame.0080
alg-3-frame.0140
alg-3-frame.0160
Alg-3-final

Fig. 5.

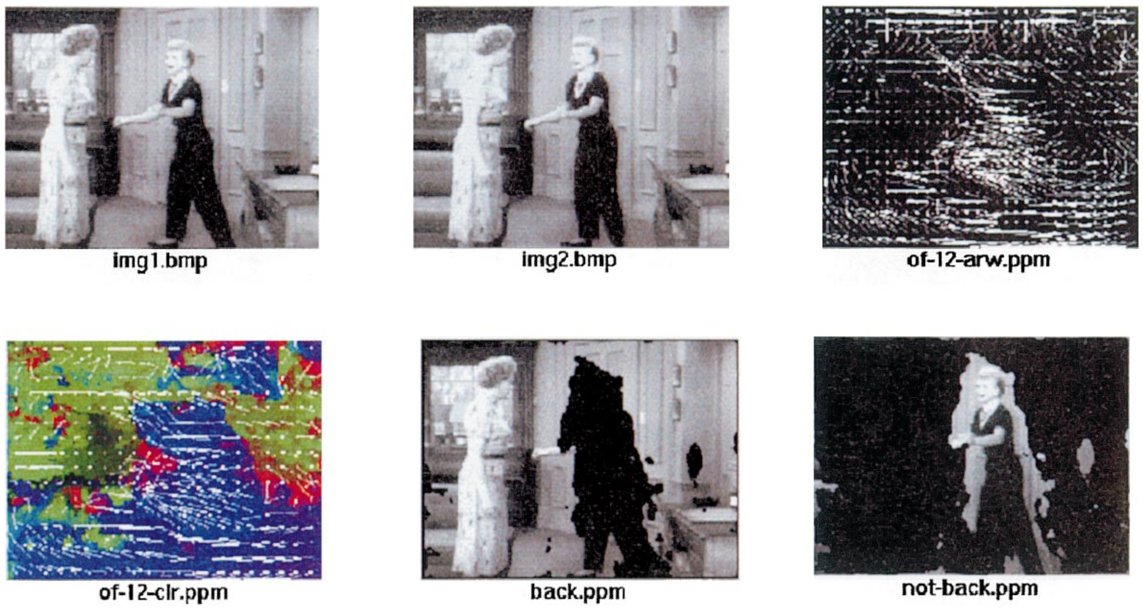
img1.bmp
img2.bmp
of-12-arw.ppm
of-12-clr.ppm
back.ppm
not-back.ppm

Fig. 6.

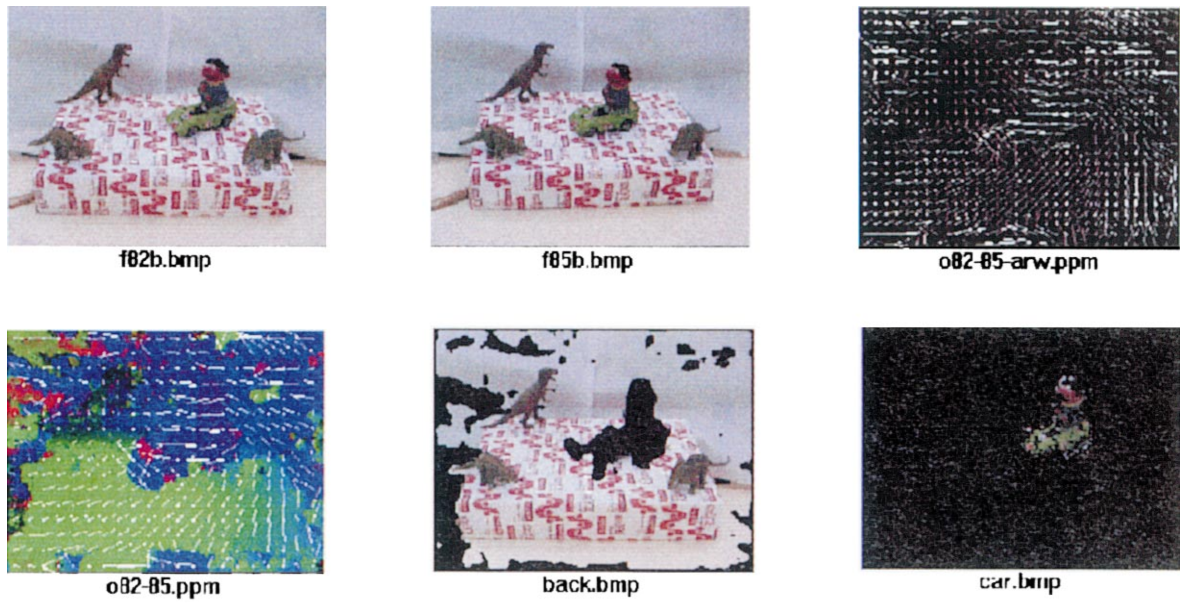

Fig. 7.

References

[1] A. Rosenfeld, A.C. Kak, Digital Picture Processing, Academic Press, New York, 1982 (Two volumes; 1st ed., Academic Press, New York, 1976. Image Processing, Survey).

[2] M. Sonka, V. Hlavac, R. Boyle, Image Processing, Analysis, and Machine Vision, Chapman and Hall, London, 1993.

[3] E. Charles A. Poynton Chiarello, J.M. Jolion, C. Amoros, Regions growing with the stochastic pyramid: application in landscape ecology, PR(29), No. 1, January 1996, pp. 61–75.

[4] A. Baraldi, F. Parmiggiani, Single linkage region growing algorithms based on the vector degree of match, GeoRS(34), No. 1, January 1996, pp. 137–148.

[5] E. Pauwels, P. Fiddelaers, L. VanGool, Grouping of contour-segments using an adaptive region-growing algorithm, ICPR96(B8E.6), 9608(ESAT, B).

[6] http://www.inforamp.net/p̃oynton/notes/colour_and_gamma/Color-FAQ.html

[7] S.S. Beauchemin, J.L. Barron, The computation of optical-flow, surveys, 27 (3) (1995) 433–467.

[8] B.D. Lucas, T. Kanade, An Iterative image registration technique with an application to stereo vision, DARPA81(pp. 121–130), and IJCAI81(pp. 674–679). Uses differences in intensity between the two images and the local gradient of one image (both?) to compute the shift. A registration problem, but very applicable to stereo.