

Hierarchical Morphological Segmentation for Image Sequence Coding

Philippe Salembier and Montse Pardàs

Abstract—This paper deals with a hierarchical morphological segmentation algorithm for image sequence coding. Mathematical morphology is very attractive for this purpose because it efficiently deals with geometrical features such as size, shape, contrast, or connectivity that can be considered as segmentation-oriented features. The algorithm follows a Top-Down procedure. It first takes into account the global information and produces a coarse segmentation, that is, with a small number of regions. Then, the segmentation quality is improved by introducing regions corresponding to more local information. The algorithm, considering sequences as being functions on a 3-D space, directly segments 3-D regions. A 3-D approach is used to get a segmentation that is stable in time and to directly solve the region correspondence problem.

Each segmentation stage relies on four basic steps: simplification, marker extraction, decision, and quality estimation. The simplification removes information from the sequence to make it easier to segment. Morphological filters based on partial reconstruction are proven to be very efficient for this purpose, especially in the case of sequences. The marker extraction identifies the presence of homogeneous 3-D regions. It is based on constrained flat region labeling and morphological contrast extraction. The goal of the decision is to precisely locate the contours of regions detected by the marker extraction. This decision is performed by a modified watershed algorithm. Finally, the quality estimation concentrates on the coding residue all the information about the 3-D regions that have not been properly segmented and therefore coded. The procedure allows the introduction of the texture and contour coding schemes within the segmentation algorithm. The coding residue is transmitted to the next segmentation stage to improve the segmentation and coding quality. Finally, segmentation and coding examples are presented to show the validity and interest of the coding approach.

I. INTRODUCTION

ONE of the most popular techniques for high compression of visual data follows the so-called “second generation image coding” approach [6]. This approach consists of taking into account the characteristics of the human visual system to define the coding scheme. A rather high number of second generation coding techniques have been proposed in the past. One of the most promising ones relies on a contour-texture approach [5], which leads to a segmentation-based coding system involving three steps. The first one is the segmentation that splits the original data into various homogeneous components corresponding as much as possible to semantic

units. The second step, called “contour coding,” consists of coding the information about the partition of the space. Finally, the last step deals with the information inside each region. It generally involves a gray level or color function describing the “texture.” The coding procedure takes into account the sensitivity difference of the human visual system with respect to contours and textures. This segmentation-oriented approach contrasts with more classical pixel-based approaches [4]. In pixel-based techniques, performances at high compression are limited because the processing does not take into account the objects geometry or the discontinuities. Segmentation-based approaches have proved to be very useful for still images [6]; however, they are even more efficient for sequences because they handle precisely the difference between spatial and temporal discontinuities or correlation [11], [20].

The present study is concerned with a segmentation-oriented approach to image sequence coding relying on morphological tools [16], [17]. Mathematical morphology is indeed very attractive for this purpose because it is a geometrical approach to signal processing and easily deals with criteria such as shape, size, contrast, connectivity, etc. Moreover, morphological transformations can be very efficiently implemented in both software and hardware. This point is of prime importance because the major bottleneck in segmentation-based coding schemes is the complexity and the computational load of the segmentation step. In this study, the main focus of interest is the segmentation step, but as will be seen in the sequel, the segmentation algorithm leads to the entire codec structure.

The algorithm described here is an extension of the approach proposed in [15], which was originally developed for still images. In [15], the usefulness of mathematical morphology for segmentation-based coding has been shown to rely on three major points: first, *morphological filters by reconstruction* are extremely useful for segmentation because they simplify the data producing flat zones without corrupting the contour information. Second, the morphological approach to segmentation consists of separating the feature extraction step, which has to locate the *presence* of homogeneous regions, from the decision, which precisely *locates* the contours of the regions. This two-step procedure gives a large degree of freedom and flexibility in the segmentation design. Third, the decision step can be very efficiently performed by the so-called *watershed* algorithm. The extension of the algorithm for still image to sequence is, in a first step, done by considering that the signal is not defined on a 2-D space but on a 3-D space. A 3-D approach is used to get a segmentation that is stable in time and to directly solve the region correspondence problem.

Manuscript received March 29, 1993; revised January 24, 1994. This work was supported by the CICYT of the Spanish government. The associate editor coordinating the review of this paper and approving it for publication was Prof. Dr.-Ing. Bernd Girod.

The authors are with the Department of Signal Theory and Communications, E.T.S.E.T.B.-Universitat Politècnica de Catalunya, Barcelona, Spain.
IEEE Log Number 9402256.

However, this approach leads to theoretical as well as practical problems [10]. The major theoretical problem concerns the image sequence itself, which cannot be simply considered as a 3-D signal. Indeed, the time axis does not play the same role as the spatial axis. Moreover, motion creates 3D components which have specific geometry and connectivity. To solve this set of problems, we will see that morphological filters with *partial* reconstruction constitute an efficient approach. On the practical side, the major concern deals with the computational complexity and the volume of data to be processed. As a consequence, we will propose a region-based algorithm in place of the contour-based algorithm of [15] which implies to work on interpolated signals. This choice will result in the use of new marker extraction techniques and in a redefinition of the watershed algorithm similar to that of [9].

The segmentation algorithm is hierarchical following a Top-Down procedure. In other words, it is a purely splitting process. Good results can be achieved because of the use of specific morphological filters, based on reconstruction processes, allowing the simplification of the signal by retaining global information while preserving the contour information. As a result, the first segmentation level produces a segmentation which is coarse in the sense that it contains a few regions, however the contours are perfectly localized. Then, the remaining steps only introduce new regions without modifying the previous segmentation result. In the framework of coding, this approach is particularly interesting for progressive transmission or storage: from the first level of the hierarchy, one can derive a coarse coded sequence composed of a few regions, which will lead to moderate visual quality but very high compression ratio. This first coded sequence can then be refined by introducing the regions segmented by the lower levels of the hierarchy. As a result, the visual quality will increase at the expenses of the compression ratio.

The organization of this paper is as follows: the next section discusses the basic hierarchical structure. Section III presents the morphological tools for the algorithm. Special attention is paid to notions and issues of interest for sequences. The basic steps of the codec structure are analyzed in detail in Section IV. Finally, Section V is devoted to the presentation of segmentation and coding results.

II. HIERARCHICAL SEGMENTATION FOR CODING

As discussed above, the segmentation and coding are achieved by successive steps. All levels of the hierarchy have basically to perform a segmentation, and therefore, they involve the same elementary steps described in Fig. 1: simplification, marker extraction, and decision and quality estimation. Let us briefly give their functional descriptions (see [13] and [15] for more details):

- The simplification goal is to make the signal easier to segment. It controls the nature and amount of information that is kept for segmentation at this level. It actually defines the notion of hierarchy. We will comment later on the meaning of the input signal called "coding residue."

- The marker extraction is used to assess the local homogeneity. It makes use of the simplified sequence and of the information about the current segmentation. Markers indicate the presence of 3-D homogeneous areas. They identify the regions interior and do not intend to solve the problem of contour localization. This is the purpose of the third step called decision.
- The decision step needs two different inputs: a signal to segment (in our case, the original data) and the set of markers defining the presence of objects. The output is the segmentation result that is a label sequence (sequence whose gray-level values correspond to the partition classes). An individual and arbitrary number is assigned to each region.
- The quality estimation indicates to the following hierarchical level where the segmentation has to be improved or refined. In our case, the segmentation goal is coding. This means that the quality estimation should indicate the areas that are not properly coded. This estimation can be achieved by actually coding each region, that is, by generating the sequence that will be reconstructed in the receiver side, and by computing the difference between the coded image and the original one. A high difference indicates the presence of a region that is poorly represented by the current segmentation and should be better represented by the next levels. By contrast, a low difference characterizes a region that has been well represented by the current coding process. Let us call this difference **the coding residue**. This process is illustrated in Fig. 1, where the coding block makes use of the new label image and the original image to compute an estimate of the coded sequence, which, in turn, is used to calculate the coding residue.

The whole hierarchy is made of a succession of schemes as the one illustrated in Fig. 1. The information that is transmitted between two levels is the coding residue, the current segmentation result (Label sequence), and the original sequence. At the very first level, the coding residue is defined as the original sequence itself, and the segmentation result is composed of a single region. The progressive or multilevel approach of the entire scheme relies on the simplification step that controls the amount of information kept for the segmentation. At the first level, this simplification is very strong and discards the major part of the information. Then, the simplification is progressively reduced to get a more precise result. Because it constitutes the heart of the multiresolution representation, an important part of this work is dedicated to the study and selection of a proper simplification tool.

The segmentation structure proposed in [15] was defined in the context of still image coding. Our objective in this paper is to extend it to image sequences. This extension is done by considering that the signals are not defined on a 2-D space but on a 3-D space with the time axis playing the role of the third dimension. Note that no difference is made between interlaced or progressive sequences. This distinction only affects the geometry of the working space: a progressive sequence can be considered as a function from a 3-D space sampled on a cubic grid, whereas an interlaced sequence is a function from

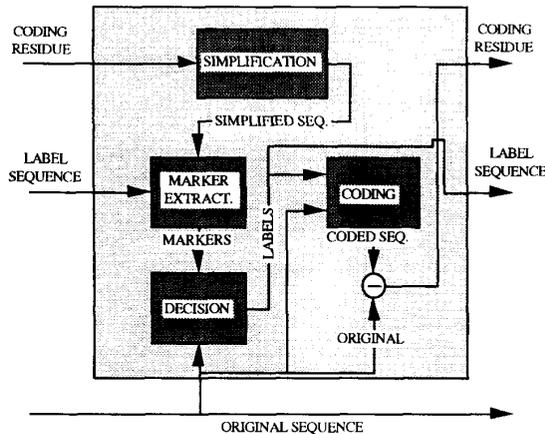


Fig. 1. Hierarchical structure for segmentation and coding.

a 3-D space sampled on an octahedral grid. The geometry of the sampling grid only modifies the definition of the local neighborhood, but all morphological transforms remain the same. Let us now briefly review some morphological tools of interest for the algorithm.

III. MORPHOLOGICAL TOOLS FOR HIERARCHICAL SEGMENTATION

The goal of this section is to briefly describe some morphological tools of interest for the algorithm. A complete description about mathematical morphology can be found in [16] and [17]. This presentation is divided into three parts. The first one is devoted to morphological operators and filters that will be mainly useful for the simplification step. The second part concentrates on morphological gradients and discusses why they should be avoided in the case of sequences. Finally, the last part deals with the decision tool known as the watershed algorithm.

A. Morphological Operators and Filters

A large number of morphological tools relies on two basic sets of transformations known as erosions and dilations. In this study, two sets of erosions and dilations are used. The first one deals with erosion and dilation with flat structuring element.

Definition 1: If $f(x)$ denotes an input signal and M_n a window or flat structuring element of size n , the erosion and dilation by the flat structuring element M_n are given by

$$\begin{aligned} \text{Erosion: } \quad \varepsilon_n(f)(x) &= \text{Min}\{f(x+y), y \in M_n\}. \\ \text{Dilation: } \quad \delta_n(f)(x) &= \text{Max}\{f(x-y), y \in M_n\}. \end{aligned} \quad (1)$$

The second set of erosion and dilation involves geodesic transforms [13]. They are always defined with respect to a reference function r .

Definition 2: The geodesic dilation of size one (that is, the smallest size on the discrete space) is defined as the minimum between the dilation of size one of the original function (f)

and the reference function (r). The geodesic erosion is defined by duality:

$$\text{Geodesic dilation of size one: } \delta^{(1)}(f, r) = \text{Min}\{\delta_1(f), r\} \quad (2)$$

$$\text{Geodesic erosion of size one: } \varepsilon^{(1)}(f, r) = -\delta^{(1)}(-f, -r).$$

Geodesic dilations and erosions of arbitrary size are defined by iterations. For example, the geodesic dilation (erosion) of infinite size, which is also called *reconstruction by dilation (by erosion)* is given by

Reconstruction by dilation:

$$\gamma^{(rec)}(f, r) = \delta^{(\infty)}(f, r) = \dots \delta^{(1)}(\dots \delta^{(1)}(f, r) \dots, r) \quad (3)$$

Reconstruction by erosion:

$$\varphi^{(rec)}(f, r) = \varepsilon^{(\infty)}(f, r) = \dots \varepsilon^{(1)}(\dots \varepsilon^{(1)}(f, r) \dots, r).$$

The choice of the basic dilation of size one defines the notion of connectivity and neighborhood. In 2-D spaces with squared sampling grid, the classical choices are the cross, which leads to 4-connectivity or the square (3×3), which leads to 8-connectivity. In 3-D spaces with a cubic sampling grid, the corresponding examples are the cross and the cube. They respectively lead to 6 or 26-connectivity. In the sequel, when a reconstruction process will be used, it will be based on 6-connectivity. Finally, reconstruction processes can be implemented very efficiently by using queues that avoid any iterating process and lead to extremely fast algorithms (see [18]).

Elementary erosions and dilations allow the definition of morphological filters such as the morphological opening and closing:

Morphological opening:

$$\gamma_n(f) = \delta_n(\varepsilon_n(f)) \text{ denoted by } \gamma_n = \delta_n \varepsilon_n \quad (4)$$

Morphological closing:

$$\varphi_n(f) = \varepsilon_n(\delta_n(f)) \text{ denoted by } \varphi_n(f) = \varepsilon_n \delta_n.$$

A morphological opening (resp. closing) simplifies the original signal by removing the bright (resp. dark) components that do not fit within the structuring element. If the simplification has to deal with both bright and dark elements, an open_close ($\gamma_n(\varphi_n(f))$) or a close_open ($\varphi_n(\gamma_n(f))$) has to be used. None of these filters are self-dual, but in practice, they approximately remove the same kind of information. Note that these filters define the notion of size by reference to their structuring elements. A 3-D component is said to be small if it does not fit within the structuring element for instance a parallelogram. As a consequence, the size of a real object in the scene depends on its motion. These filters can be used as simplification tool before segmentation, but they do not allow a perfect preservation of the contour information [12]. In order to improve the contour preservation properties filters by reconstruction can be used.

The first filter by reconstruction that will be used in the sequel is the opening by reconstruction of erosion or opening (of course, by duality a closing can be defined):

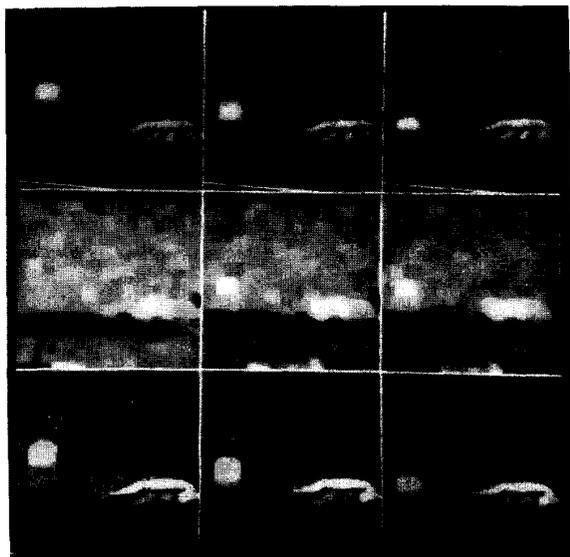


Fig. 2. Simplification with morphological filters: first row: original sequence, second row: open_close filter, third row: open_close by reconstruction.

Opening (closing) by reconstruction of erosion (dilation):

$$\gamma^{(\text{rec})}(\varepsilon_n(f), f), \varphi^{(\text{rec})}(\delta_n(f), f). \quad (5)$$

In the case of $\gamma^{(\text{rec})}(\varepsilon_n(f), f)$, the simplification is performed by the erosion that eliminates the bright components that are smaller than the structuring element. Then, the reconstruction process restores the contour of the components that have not been totally removed by the erosion. Fig. 2 shows a sequence with the results of a morphological open_close and an open_close by reconstruction. In both cases, the structuring element is a parallelogram of size $31 \times 31 \times 3$ (spatial \times spatial \times temporal). Intuitively, images obtained with filters by reconstruction seem to be a good starting point for the segmentation. They are much simpler than the original images, but the objects that are present are precisely defined. However, one can see a problem that is typical of moving objects: The part of the background (the wall) appearing on the trajectory of the moving ball is noisy. In fact, the reconstruction process has artificially connected some texture points of the wall on the second (respectively third) frame with the ball in the first (respectively second) frame. This phenomenon is typical of sequences processed as 3-D signals since covered and uncovered areas may be connected during the reconstruction process.

To solve this problem, one can consider that it is not appropriate to reconstruct totally the contour, especially in the time direction. To this end, the reference signal for the reconstruction process, that is f , may be smoothed by a small opening $\gamma_k(f)$. This approach leads to *partial reconstruction* filters [14]:

Opening, closing by partial reconstruction:

$$\gamma^{(\text{rec})}(\varepsilon_n(f), \gamma_k(f)), \varphi^{(\text{rec})}(\delta_n(f), \varphi_k(f)). \quad (6)$$

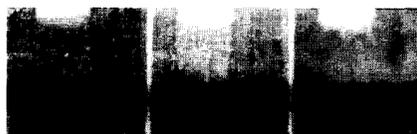


Fig. 3. Partial versus full reconstruction: First image: original frame, second image: open_close by reconstruction, third image: open_close with partial reconstruction.

The parameter k allows a smooth tuning of the reconstruction from no reconstruction ($k = n$), that is, morphological opening or closing, to "full" reconstruction ($k = 0$). To avoid any artificial temporal connection, the reference signal can be simply a *spatial* opening by reconstruction of small size of the original signal: $\gamma^{(\text{rec})}(\varepsilon_{n1}(f), f)$. Spatial opening means that the erosion and the reconstruction are considered as intraframe processing. Fig. 3 presents a zoomed part of the sequence. These images have been normalized to allow a good visualization. The first image is a frame of the original sequence, the second and third images, respectively, show the results of an open_close by reconstruction $\gamma^{(\text{rec})}(\varepsilon_{31 \times 31 \times 3}(f), f)$, and by partial reconstruction, $\gamma^{(\text{rec})}(\varepsilon_{31 \times 31 \times 3}(f), \gamma^{(\text{rec})}(\varepsilon_{3 \times 3}(f), f))$. The advantage of using partial reconstruction is clearly visible. This subjective quality assessment will be confirmed in Section IV-B by objective measures.

B. Morphological Gradients

The classical morphological approach to segmentation relies on gradients. Let us make a few comments about gradients and why they should not be used in the case of sequences. In morphology, three gradients are generally used:

Morphological gradient:

$$g = \delta_1(f) - \varepsilon_1(f)$$

Gradient by erosion, by dilation:

$$g^- = f - \varepsilon_1(f), \quad g^+ = \delta_1(f) - f. \quad (7)$$

All gradients are positive, but the first one is symmetrical with respect to the contour position, whereas the two remaining ones are not. In [15], it was mentioned that the use of the gradient results in a loss of information. In particular, if the original signal involves transitions, its gradient is either biased (gradient by erosion or dilation) or thick (2 pixels). In the case of still images, this phenomenon is not extremely annoying. However, for a precise segmentation, an interpolated gradient should be used [15]. In the case of moving images, the use of the gradient results in a much larger loss of information [10]. For instance, Fig. 4 shows a moving circle and its gradient: the thickness of the gradient depends on the motion of the object: Without motion, the gradient is two pixels wide; however, this thickness increases with motion. It is impossible to find the contours of the circle only from its gradient (note that the first and last frames of the gradient are not equal to the central frame because frames before the first and after the last frames are supposed to be black).

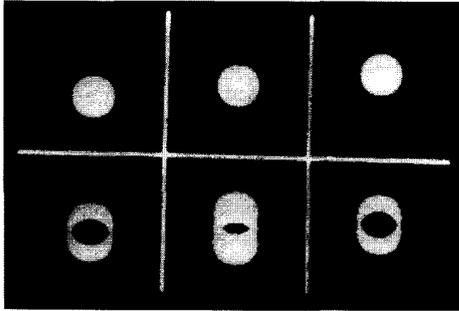


Fig. 4. Loss of information of the gradient: First row: Original sequence of moving circle; second row: morphological gradient.

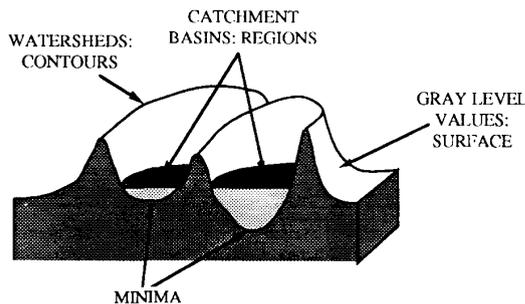


Fig. 5. Watershed algorithm.

C. The Watershed Algorithm

The watershed algorithm derives from topographic works where the *catchment basins* and their dividing lines, which are called *watershed lines*, have been extensively studied in the past (see [19]). In image processing, this notion has been introduced by considering the gray-level values of a picture as the altitude of an imaginary relief. As can be seen in Fig. 5, watershed lines partition the space by associating a region called a *catchment basin* to each local minimum. A large number of algorithms have been proposed for the efficient computation of watershed. The most efficient ones are based on immersion simulations [19] and rely on hierarchical queues [9], [19].

Immersion simulation consists in flooding the surface from its local minimum. Starting from the minimum of lowest altitude, the water progressively fills up the catchment basins. When the water level reaches the altitude of other minima, these minima start to be active, and the flooding process also originates from these minima. Now, when the water coming from two different minima would merge, an imaginary dam is built to prevent any mixing of water. The procedure is ended when the water level is higher than the absolute maximum. In this case, each minimum is surrounded by water, that is, its catchment basin, and a dam delimiting its border, that is, its watershed line. The catchment basins constitute a partition of the space. It is, in fact, a segmentation based on the signal minima. Note that the immersion simulation relies on a double ordering: First, a point at a given altitude will be flooded after the points of lower altitude, and second, at a given altitude,

points that are close to some flooded points will be flooded before points that are far away. Let us describe the algorithm implementation with hierarchical queues.

Efficient implementations of the immersion simulation require a clever scanning. Indeed, if classical raster scanings are used, the resulting implementations are extremely inefficient because a very large number of pixels are only examined without being able to decide if they belong to a particular catchment basin or to a watershed line. The idea of queue-oriented algorithms is to redefine the scanning procedure in such a way that it is always possible to make a decision about the current pixel. That is to say to which catchment basin it belongs or if it is a watershed line point.

A hierarchical queue is a set of queues with different priorities [9], and each queue is a first-in-first-out (FIFO) data structure. The elements processed by the queue are the pixel positions (the queue is used to define the scanning). This structure allows the representation of a double ordering: Pixels are put into one of the queues depending on a notion of priority. The first pixel to be pulled out of the queue is the first one that has entered the queue of highest priority. Then, successively, all pixels in the queue of highest priority are extracted. Finally, if the queues of highest priority are empty, the first pixel to be extracted is the first pixel of the first nonempty queue. Note that this procedure differs slightly from the one described in [9] in the sense that when a queue of high priority is empty, it is not removed and pixels of high priority are allowed to come later and fill these empty queues. The procedure is illustrated by Fig. 6. As can be seen, the first (inner) order is constituted by the order of the pixels inside a given queue, and the second (outer) order is represented by the priority of the queues themselves.

Now, the immersion algorithm can be simply implemented with these queues. The algorithm works in two distinct steps: queue initialization and flooding procedure. The initialization consists of putting the locations of all signal local minima in the queue with the opposite of their gray-level value as priority. Thus, the queue of highest priority corresponds to the pixels of absolute minimum. The flooding consists in extracting a pixel from the queue: If the pixel does not yet belong to a catchment basin, we know, because of the filling procedure, that it has at least one neighbor belonging to a catchment basin. Therefore, all neighbors of the current pixel belonging to a catchment basin are examined, and the pixel is assigned to the catchment basin corresponding to the neighbor of closest gray-level value. Then, if the current pixel has some neighbors that do not belong to any catchment basin, these neighbors are put in the queue with a priority defined as the inverse of the gray-level value. Note however, that one should check that those pixels are not already in the queue. Indeed, these pixels can be neighbors of pixels examined previously and may have already been put in the queue. As one can see, any pixel that is put in the queue has at least one of its neighbors belonging to a catchment basin. This is why it will be possible to make a decision concerning this pixel when it will come out of the queue. The flooding procedure is illustrated in Fig. 6 in the case of 2-D images. In the case of 3-D signals, that is, for image sequences, the procedure is exactly the same, and the

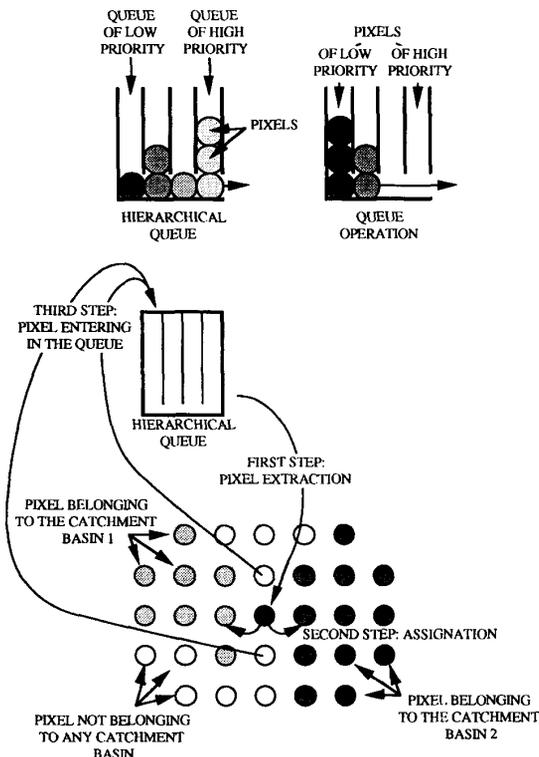


Fig. 6. Implementation of the watershed algorithm with a hierarchical queue.

only difference is the definition of the neighborhood (as in the case of reconstruction).

The watershed algorithm is one of the major decision tools in mathematical morphology. To get the contours of objects, the watershed should work on the morphological gradient of the signal to segment. Indeed, a contour in the signal corresponds to a bright line in the morphological gradient. However, the direct segmentation of the gradient by the watershed results in an extreme oversegmentation. The algorithm assigns a different region for each individual local minimum of the gradient. To solve this problem, the gradient can be simplified to segment only the objects of interest that are the objects that have been "marked" in the feature extraction step. Refer to [8] and [19] for more details. We do not give more explanations concerning this approach because, as discussed in Section III-B, our algorithm will avoid the use of the gradient.

IV. ELEMENTARY SEGMENTATION STEPS

A. Sequence Simplification

The heart of the hierarchical approach relies on the simplification filter. Indeed, it controls the type and amount of information that is removed from the sequence before marker extraction and decision. As a consequence, it defines the notion of hierarchy in the segmentation result and, ultimately, the various levels in coding quality or compression ratio.

This section is devoted to the study and selection of the simplification filter. In a first step, a criterion will be defined. Then, the performance of the filters presented in Section III-B will be assessed and discussed.

1) *Simplification Criterion*: Assume that a gray-level segmentation algorithm is used after simplification. This restriction is mainly done to avoid very complex and time-consuming texture segmentation techniques. With this assumption, an easy signal to segment is a signal composed of constant gray-level regions with sharp contours corresponding precisely to those of the original signal. To quantitatively assess the quality of filters presented in Section III, synthetic test sequences have been used. They are composed of moving objects (triangles, polygons, etc.) on a constant background, and they are corrupted by noise. The advantage of using a synthetic sequence is that the optimal segmentation result is *a priori* known. The test sequences are first simplified by a morphological filter and then segmented (to have coherent results, the segmentation relies on the watershed algorithm to be described in the sequel). Once a test sequence has been segmented, two parameters are measured:

- **Edge Localization**: This parameter is defined as the number of pixels differing between the current and the optimal segmentation divided by the area of the objects to segment. It measures the contour preservation property of the filter.
- **Flatness**: This parameter is the variance of the simplified signal inside each segmented region. It assesses the filter efficiency to produce flat, and therefore easily segmentable, regions.

Each measure is plotted on a 2-D (edge localization/flatness) plane. For each filter, a set of measures is obtained by modifying the structuring element size. They create a curve in the edge localization/flatness plane. For the results reported in the following, only the spatial size has been modified while the temporal size has been fixed to 3. Note that this procedure is similar to the one reported in [14]. The main difference is the flatness criterion used here because the filters are studied in the context of segmentation.

Finally, several tests have been performed with various test sequences and noise probability density functions such as Laplacian, Gaussian, and uniform. As far as the criteria described above are concerned, they lead to very similar results. Therefore, only the result obtained with a specific sequence and Gaussian noise will be presented here.

2) *Simplification Performance*: Fig. 7 presents the performance of a median and three morphological filters. Ideally, a good simplification filter has low edge localization and flatness parameters. Its curve should lie close to the origin of the plane. It can be seen that the worst filters are the median and morphological open_close filters. They produce a relatively poor flatness, and their contour preservation becomes quickly bad. The points on the left (resp. right) side of the curve represent the filter performances for small (resp. large) structuring elements. The same experiments were done with linear low-pass filters that give extremely bad results. The morphological open_close by reconstruction achieves a much

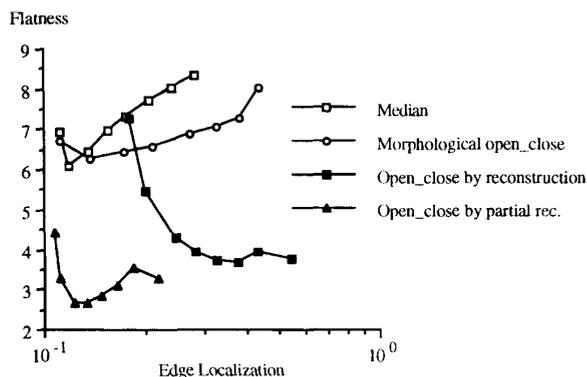


Fig. 7. Flatness versus edge localization of simplification filters.

better flatness but at the expense of the contour information. A precise study of the simplified images reveals that this degradation results from the artificial temporal connections created by the reconstruction process. Finally, the best filter for both criteria is the open_close by partial reconstruction, which produces flat regions with well-localized contours. The partial reconstruction is obtained by using as reference signal the result of a small (3×3) spatial opening by reconstruction.

Fig. 8 presents similar results but with the *alternating sequential* version of these filters: Let us recall the definition of alternating sequential filters: If Ψ_n denotes a morphological filter of the type open_close involving a structuring element of size n , its alternating sequential version is given by

$$\text{Alternating sequential filter of } \Psi_n : \\ \Psi_n(\Psi_{n-1}(\dots \Psi_k \dots (\Psi_1(\dots)))) \quad (8)$$

These filters can also be viewed as a multiresolution approach to the filtering problem [14]. As far as the relative performances are concerned, the conclusions are similar to that of the previous experiment. On the average, the results are better in the case of alternating sequential filters, and the major improvement concerns the region flatness. In fact, the first filtering stages with small structuring elements progressively remove small noise components that would otherwise be responsible for some fluctuations after filtering. The open_close by partial reconstruction still exhibits the best performance for both parameters. In this case, the major advantage of using the alternating sequential approach is the robustness regarding the structuring element size.

These experiments allow the selection of simplification filters for segmentation. For a general-purpose algorithm, a good complexity/performance scheme is based on open_close by partial reconstruction with a simplified alternating sequential version, for instance, $\Psi_n(\Psi_{n/2}(\dots))$. In the following, this filtering scheme is assumed to be used for simplification. This conclusion differs from the one reported in [15] for still images where the use of partial reconstruction was not so attractive. In the case of moving sequences, partial reconstruction has a much higher interest.

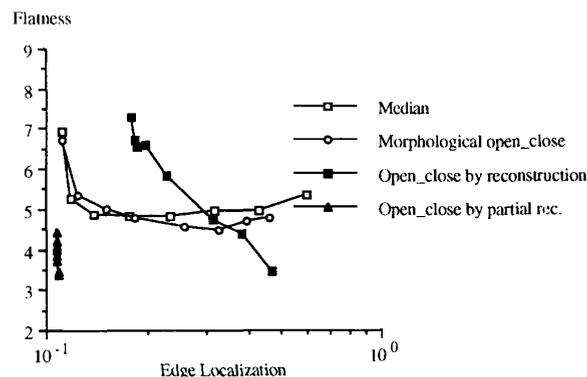


Fig. 8. Flatness versus edge localization of simplification alternating sequential filters.

B. Marker Extraction

As discussed in [8], a very efficient morphological approach to segmentation relies on “marker extraction” followed by the watershed algorithm. We will follow this approach but will avoid the use of the gradient. Indeed, as discussed in Section III, the use of gradient results in a severe loss of information in the case of moving objects and the solution proposed in [15] of using interpolated signals is not practical in the case of sequences. The output of the marker extraction step is a set of “labeled markers.” Labeled markers are gray-level signals (sequences in our case) identifying the presence of homogeneous 3-D regions that will be precisely delimited by the decision step. The interior of each homogeneous region is “marked” by a label, that is, a constant gray level value, which is unique for this region. Moreover, the zones that are not homogeneous or in between two homogeneous regions are not “marked,” and a zero label is associated with them. They represent uncertainty areas.

The marker extraction has to make use of several sources of information to identify homogeneous regions. The first one is the current segmentation. Indeed, the previous segmentation levels have produced a current segmentation result that may not be very complete but already defines some homogeneous regions. Note that the segmentation result can be viewed as a labeled marker without uncertainty areas. The second source of information is the simplified image (see Fig. 1). Several techniques can be jointly used to extract the sets of markers.

Let us describe the marker extraction from the simplified sequence. It has to rely on the features of the simplification. As shown previously, open_close filters with partial reconstruction simplify the signal by producing flat regions while retaining the edge information. This remark leads us to use two techniques: one extracting flat regions and another one looking for high contrast. The first marker extraction is based on “constrained flat region labeling,” whereas the second one relies on residues of morphological centers.

1) *Marker Extraction for Flat Regions:* This first marker extraction technique aims at finding flat regions in the simplified image. These regions can very simply be identified by labeling flat regions, that is, by labeling the connected

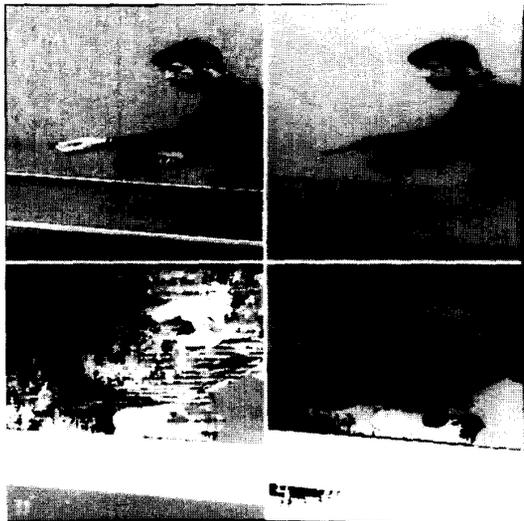


Fig. 9. Example of labeling of flat regions: First row: original and simplified image, second row: labeling of all flat regions (3540 regions) and labeling of flat regions larger than 1000 pixels (eight regions).

components of the space where the function is of constant gray-level value. Let us briefly describe the labeling algorithm.

As in the case of the watershed, a very efficient implementation of a labeling algorithm uses a queue structure (hierarchical queue with one level of hierarchy): The first pixel of the sequence is put in the queue as well as its neighbors (3-D neighbors in the case of sequences) that have the same gray-level value. When a pixel is extracted from the queue, because of the filling procedure, we know that this pixel has at least one neighbor of same gray-level value that has already a label. This label is assigned to the current pixel. Then, all its neighbors of the same gray-level value without a label are put in the queue, and the procedure is iterated until the queue is empty. When the queue is empty, it means that the current flat zone has been entirely labeled; therefore, the label number is increased, and the first nonlabeled pixel of the signal is searched and put in the queue. Finally, the sequence has been labeled when all pixels have a label. Note that a flat region can very well be composed of a single pixel.

This labeling algorithm has to be modified for two reasons: first, the labeling should take into account the current segmentation result, that is the segmentation performed by the previous hierarchical segmentation levels. Second, the flat regions of interest are not all flat regions but, because of the simplification filter, flat regions of size larger than a minimum. Indeed, the simplification filter has removed signal components smaller than a given limit. Therefore, flat regions of size smaller than this limit are not of interest and constitute uncertainty areas, that is transition pixels between two large flat zones. As a result the labeling algorithm has to check not only that the current pixel and its neighbors are of the same gray level value (that is the local flatness) but also that they correspond to the same region with respect to the current segmentation. Moreover, once a flat region has been labeled, if its size (number of pixels) is smaller than

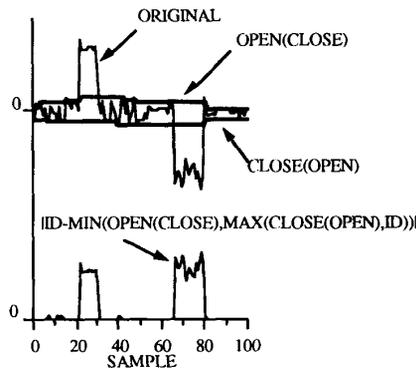


Fig. 10. Contrast extraction.

the minimum size defined by the simplification, its label is removed and the region is considered as uncertainty area. This labeling algorithm can be viewed as a constrained labeling algorithm. As an illustration, the upper part of Fig. 9 shows an original frame and its simplification by an open_close by partial reconstruction. The lower left image gives the result of the labeling of all flat regions. The number of regions is equal to 3540. Finally, the lower right image gives the labels of all regions larger than 1000. There are only eight such regions. The regions without labels are represented in black.

This simple marker extraction technique gives good results; however, it can be completed by another technique. Indeed, in the lower levels of the hierarchy, the simplification filter only removes very small components, and the size criterion used in the labeling algorithm is not pertinent. A more useful information on the lower levels is the contrast of the residue:

2) *Marker Extraction for Contrast Regions*: The marker extraction based on contrast [15] can be achieved by computing the difference between the identity and the morphological center of Id , $\gamma_n \varphi_n$, and $\varphi_n \gamma_n$:

Contrast extraction:

$$|Id - \text{Min}\{\gamma_n \varphi_n, \text{Max}\{\varphi_n \gamma_n, Id\}\}|. \quad (9)$$

This transform is illustrated in Fig. 10, where it can be seen that the open_close ($\varphi_n \gamma_n$) and close_open ($\gamma_n \varphi_n$) filters generate upper and lower noise envelopes that allow a very reliable marker extraction. The contrast-oriented marker extraction uses opening and closing by reconstruction to obtain markers whose shapes are close to the real ones. The size of the structuring element is chosen in accordance with that of the simplification step. Indeed, the simplification at level k should remove all components of size smaller than its structuring element. Let us denote by n_k this size. The marker extraction works on this signal representing objects of size larger than n_k . Therefore, the structuring element of the marker extraction should be larger than n_k . However, in the previous level of the hierarchy $k-1$, objects of size larger than n_{k-1} ($n_{k-1} > n_k$) have been segmented. This means that there is no need to use a structuring element of size larger than n_{k-1} . The results that will be shown in the next section have been obtained by setting the size of the structuring element to n_{k-1} .

The marker extraction based on contrast information leads to a binary marker, that is, a bivalued signal indicating if the signal is in between the envelopes or not. To transform it into a labeled marker, the constrained labeling algorithm described previously has to be used. The constraint signal of this labeling is simply the result of the constrained labeling of the flat regions.

C. Decision with a Modified Watershed Algorithm

Once the markers have been defined, the decision can be taken by the watershed algorithm. The classical way of using the watershed algorithm to get the objects contours is to work on the morphological gradient of the signal to segment (see Section III-C). However, as discussed in Section III-B, the use of the gradient results in a loss of information that is especially important in the case of moving objects. Therefore, we are going to use a modified watershed algorithm working *directly* on the signal and not on its gradient. The idea of using the watershed algorithm directly on the signal to segment was proposed in [9] to deal with color images. Our approach follows these principles with some slight modifications.

With respect to the algorithm described in Section III-C, the modifications rely on two points: First, the pixels that are processed by the algorithm are not pixels of the gradient but pixels of the signal itself. Second, in the algorithm of Section III-C, the priority of a pixel was defined by its gray-level value: A high (low) priority was assigned to a dark (bright) pixel. Note that independently of the time instant a pixel is introduced in the queue, it will always have the same priority. In this modified watershed, the priority is defined as the degree of certainty with which a pixel belongs to a given region. Let us call this distance the degree of certainty. Of course, various distances and, therefore, priorities can be used. In the following, the distance is defined as the absolute difference between the pixel gray-level value and the mean gray-level value of the pixels belonging to the neighboring region. The pixel priority is the opposite of its distance to a region. As in Section III-C, the algorithm involves two steps: initialization and flooding.

- The **initialization** puts in the queue the location of all pixels corresponding to the interior of a region in the labeled marker (pixels not belonging to uncertainty areas). These pixels have the highest priority (distance 0) because they certainly belong to their respective regions.
- The **flooding** assigns pixels to regions following a region growing procedure. To constrain the current segmentation to the segmentation obtained in the previous levels, the algorithm checks that the region and the pixel under consideration are compatible, that is, if they belong to the same partition class in the previous segmentation. Two incompatible pixels should not be part of the same region. The flooding extracts a pixel from the queue. If the pixel does not belong yet to a region, we know that at least one of its compatible neighbors belongs to a region. Therefore, all compatible neighboring regions are examined, the distances between these neighboring regions and the current pixel are assessed, and the pixel

is assigned to the region giving the highest certainty. Of course, if there is only one compatible neighboring region, the pixel is directly assigned to it. Note that once a new pixel has been assigned to a region, the mean gray-level value of the region should be updated in order to accurately compute its distance with respect to new pixels. Then, if the pixel that has been assigned to a given region A has some compatible neighbors that do not belong to any region, these neighbors are put in the queue with a priority defined by their distance to the region A. Note, however, that these pixels can be neighbors of pixels previously examined and assigned. As a result they may already be in the queue *with an arbitrary priority*. This contrasts with the classical watershed algorithm where the pixel's priority is uniquely defined by its gray-level value, and if a pixel is already in the queue, it is not necessary to put it in again. Here, since the priority does not only depend on the pixel gray-level value but also on characteristics of one of its neighboring regions, its priority may change. As a result, the pixel should always be introduced in the queue again except if it is already in the queue with a higher (or equal) priority. This is the reason why, as discussed in Section III-C, if a queue of a particular priority is empty, it should not be removed so that pixels of high priority are allowed to come later (note that in the classical watershed, this distinction is useless).

D. Quality Estimation by Sequence Coding

The result of the segmentation is a label sequence defining the 3-D partition of the sequence. To have a multilevel procedure, information should be transmitted to the next level that will improve the segmentation results. The coding residue concentrates the information about the final quality of the image sequence. A low coding residue corresponds to an area that is well represented by the coding process and should not be modified. By contrast, a high coding residue represents a region that is not well coded and should be split into two or more regions to improve the coding.

The coding residue is created by actually coding the contour and the texture of each region and by computing the difference between the original sequence and the coded one. Because the goal is to represent the objects of a given size that have been extracted, a large number of coding techniques may be used. Since the algorithm involves various successive segmentation steps, the concept of working with the coding residue allows the introduction and the handling of coding models within the segmentation.

V. RESULTS

The goal of this section is to illustrate the hierarchical segmentation and coding techniques discussed in Section IV. Specific coding techniques will be used for the quality estimation. It has to be mentioned that they only constitute examples and any coding technique can be used. This is one of the important features of the algorithm that allows the introduction of the coding process in the segmentation itself.

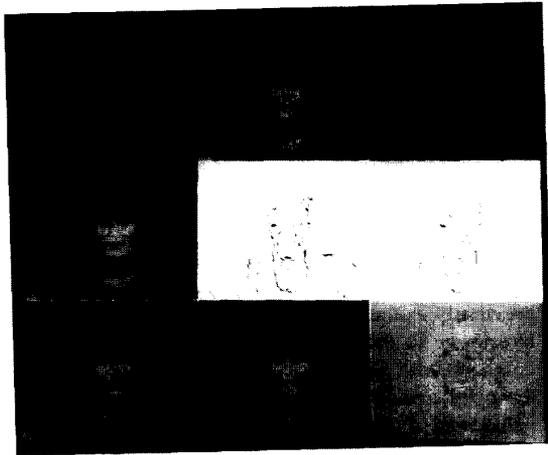


Fig. 11. Example of one step of segmentation and coding: First row—original sequence; second row: Simplified sequence, segmentation, transmitted contours; third row: coded image with low-order polynomial, coded image with BTC, coding residue.

An illustration of the segmentation and coding procedure is shown in Fig. 11. The original frame, which has been extracted from the "Miss America" sequence, is presented on the first row. In a first step, the sequence is simplified by an open-close by partial reconstruction with a structuring element of size $7 \times 7 \times 3$ (spatial \times spatial \times temporal). The result is shown in the first image of the second row of Fig. 11. Markers are extracted from the simplified sequence. As described in Section IV, the extraction works on the flatness of the regions and their contrast. From our experience, it seems that the most useful information in the higher levels of the hierarchy (when the simplification filter uses a large structuring element) is the flatness, whereas in the lower levels (when the simplification filter uses a small structuring element), the contrast is more useful. Then, the decision precisely states on the contours of the objects identified by the marker extraction. These contours are shown in the image in the middle of the second row of Fig. 11. The fourth segmentation step is the quality estimation which is the actual region coding.

First, contours are coded with the method proposed in [3]. This method starts by coding the spatial contour of the first frame in intramode by using the so-called "chain code" technique proposed in [2] and improved in [7]. It leads to an average number of 1.3 b/contour point (this figure includes both the contour description and the location of the starting points of each contour). For the following frames, the coding procedure consists of the following steps:

- Prediction of the contours of the future frame by using motion compensation
- computation of the contour prediction error
- simplification of the error with morphological tools
- transmission of the useful prediction error.

The motion information consists of one vector describing the translation of one region between two frames. On the average, the prediction error and the motion information can be coded with 0.4 b/contour point. As can be seen in the last image of

the second row of Fig. 11 showing the contours restored in the receiver, this contour coding technique is not lossless, but the loss is really small. Let us mention that motion compensation has been used for the coding step to remove as much as possible the temporal correlation. It is basically a coding of spatial contours. However, the compensation is efficient if the contours are stable in time and if the region correspondence problem has been solved. These two requirements are reached because of the use of a 3-D segmentation algorithm.

The texture within each region is first coded by low-order polynomials. An illustration of this step can be seen in the first image of the third row of Fig. 11. Then, the difference between each region and its approximation by low-order polynomials is computed. If the difference signal has a high-variance gradient, it means that the region is very active and has not been properly represented by low-order polynomials. In this case, a second texture coding technique is applied. We have used a region-oriented version of the well-known block truncation coding technique (BTC) [1]. The classical BTC takes a block of the image, thresholds it, and sends it as a binary signal using roughly 1 b/pixel. On the receiver side, the binary signal is restored with two gray levels. The threshold and restitution levels are computed so that the mean and the variance of the block be preserved by the coding. The same approach is used here but on a region basis instead of on a block basis. Moreover, it is applied on a subsampled (factor 4) version of the difference between the original image and its approximation by low-order polynomials. Since it works on subsampled signals, the BTC technique requires a little more than 0.25 b/texture pixel. Its cost is rather expensive, but first, it is used in a reduced number of regions, and second, it is very efficient to represent complex active regions that would lead to a very high number of contour points if the algorithm segments them. Here, once these regions have been coded with BTC, they will produce a low coding residue, and the following segmentation steps will not split these regions. To save bits, all frames except the first one are coded by motion compensation, that is, the texture is predicted as in the case of contours, and the prediction error is coded by low-order polynomials plus the region-based BTC described previously if necessary. The results of the coding procedure are shown on the central image of the third row of Fig. 11. In order to improve the quality of the first coded sequence, new segmentation steps are applied to the coding residue, which is presented on the right side of the last row of Fig. 11. The main difference between these following steps and the first one is the simplification involving a smaller structuring element.

Examples of hierarchical coding are shown in Figs. 12 and 13. They respectively correspond to the "Miss America" sequence in QCIF format (176 pixels \times 144 lines) and the "Table Tennis" sequence in CIF (352 pixels \times 288 lines) format. The first row of Fig. 12 shows four frames of the original sequence. The second row corresponds to the first hierarchical level and represents the coded image after segmentation in five regions and coding of contours and texture (with the mean of each region). As can be seen in Table I, apart from the first frame, the remaining frames require, on average, 210 b. If a transmission of 10 Hz of frame rate is assumed, this will lead to

TABLE I
CODING RESULTS FOR SEQUENCES OF FIG. 12

Sequence (QCIF Format)	Number of regions	First frame	Remaining frames (average figures)				Total number of Bits
		Bits	Bits for contour	Bits for motion	Bits for mean	Bits for BTC	
Second row of Fig. 12	5	1504	94	96	20	-	210
Third row of Fig. 12	23	3032	256	296	30	-	582
Fourth row of Fig. 12	23	3552	256	296	30	483	1065
Fifth row of Fig. 12	43	4632	468	488	68	512	1536

TABLE II
CODING RESULTS FOR IMAGES OF FIG. 13

Sequence (CIF Format)	Number of regions	First frame	Remaining frames (average figures)			Total number of Bits
		Bits	Bits for contour	Bits for motion	Bits for mean	
Second row of Fig. 13	3	2008	352	64	12	428
Third row of Fig. 13	9	3032	362	200	36	598
Fourth row of Fig. 13	15	3552	390	264	60	774

2,1 Kb/s. Of course, this image is of low quality, and it should be improved by another hierarchical level. The result involves 23 regions and is shown in the third row of Fig. 12. For this sequence, the texture of each region has been coded with the mean only. It leads to an average number of 582 b/frame. To further improve the quality of this sequence, two different strategies can be used: Either segment more of the image, or improve the quality of the texture coding. This second option is illustrated in the fourth row of Fig. 12, where the texture has been coded by the mean plus the region-based BTC in the active regions (mainly the face). As can be seen in Table I, on average, this sequence requires 1065 b/frame. Finally, the last row of Fig. 12 illustrates a third hierarchical step, leading to an efficiency of 1536 b/frame. As can be seen, the proposed approach is perfectly suited for very low bit rate video coding. Moreover, it is flexible both in terms of the coding techniques that are used and in terms of the quality/cost tradeoff. Note that the proposed approach makes no assumption about the sequence content. To illustrate this point, the last example shown in Fig. 13 presents a sequence that is very different from a "head and shoulder" sequence. Three hierarchical levels are shown. They, respectively, correspond to an average number of bits per frame of 428, 598, and 774. If a frame rate of 10 Hz is used, the resulting bitrates are equal to 4,2, 6, and 7,7 Kb/s. Detailed figures can be found in Table II (note that this sequence is in CIF format (352×288)).

These results show the interest of the morphological approach to segmentation-based sequence coding. Note that this scheme is particularly suitable for progressive coding since the contours of coarse segmented and coded sequences are well defined; when they appear on one level, they will remain until the last level. This result would be very difficult to obtain without morphological tools such as filters by reconstruction (see [11] or [20], for example). This approach allows the integration of various coding techniques (for contour and inside). In particular, if the modeling technique is able to

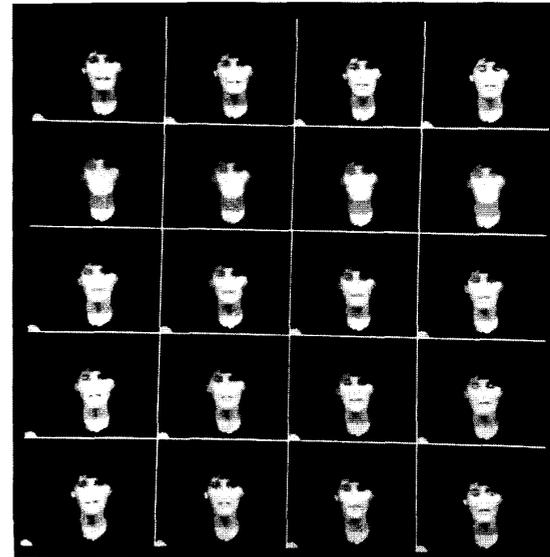


Fig. 12. Hierarchical segmentation and coding (See Table I for figures)—First row: original sequence; second row: first level (210b/frame); third row: second level with texture coding by the mean value (582 b/frame); fourth row: second level with texture coding by mean and BTC (1065 b/frame); fifth row: third level (1536 b/frame).

perfectly represent one region, by construction, this region will not be further divided by the lower levels of the hierarchy. A very important point of the algorithm is that it makes no assumption about the scene content and works for any kind of sequences. Finally, an attractive feature of the approach is its low computational complexity: For the examples presented in this section, each segmentation level requires typically 15 s of CPU per CIF frame on a regular Sparc workstation. This figure obtained without specialized hardware is very low compared with classical 3-D split-and-merge or region-

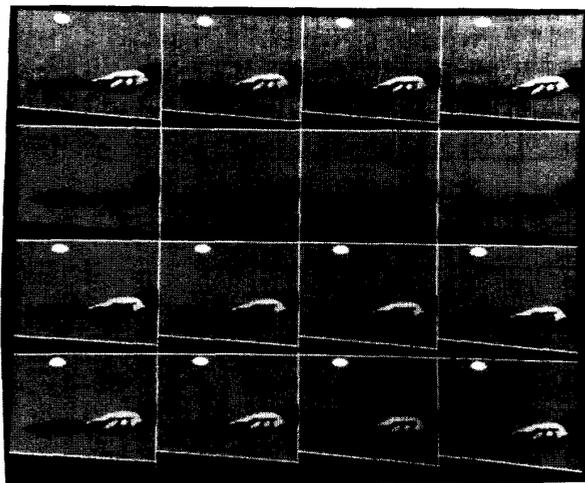


Fig. 13. Hierarchical segmentation and coding (See Table II for figures)—First row: original sequence; second row: first level (428b/frame); third row: second level (598 b/frame); fourth row: third level (774 b/frame).

growing algorithms [11], [20]. It reflects the low complexity of the algorithm.

VI. CONCLUSION

In this paper, an unsupervised hierarchical segmentation algorithm for image sequence coding based on morphological tools has been presented. It follows a purely Top-Down procedure. One of the advantages of the scheme is its low computational cost. Moreover, the approach is particularly suitable for progressive coding. The algorithm relies on a structure involving four steps that are iterated to get a multilevel segmentation: simplification, marker extraction, decision, and quality estimation.

The simplification removes part of the signal while retaining the contours information. Several filtering schemes have been compared on the basis of two segmentation-oriented criteria: flatness and edge localization. It has been concluded that the simplification for segmentation can be efficiently achieved by filters based on opening and closing by partial reconstruction. The introduction of partial reconstruction results from the apparition of problems (arbitrary temporal connection) created by the "classical" reconstruction process in the case of image sequences. The first segmentation level uses a filter with a large structuring element, keeping only the largest and more global components. On the other levels, the size of the structuring element is progressively decreased to allow the introduction of more local information to improve the segmentation.

The marker extraction takes advantage of the simplification that produces flat or contrasted regions. These two characteristics are used by two different schemes. The first one identifies flat regions larger than a minimum value by means of constrained labeling algorithm. The second scheme extracts zones of high contrast by computing the residue with a morphological center. The marker extraction output is an

image indicating the presence of homogeneous regions whose contours are not precisely defined.

The decision stating the location of the contours relies on a modified watershed algorithm working directly on the original image and not on its gradient. It has been shown that the use of the gradient results in a severe loss of contour information. The loss increases as a function of the motion of objects. To have a precise localization of the contour, a modification of the classical watershed algorithm has been proposed and discussed.

Finally, the current segmentation or coding quality is estimated by coding each region, that is, by computing the sequence that will be restored on the receiver side. Then, the difference between the original sequence and the modeled one, which is called the coding residue, is computed and transmitted to the next level. The coding residue concentrates all the information about the regions that have to be split by the segmentation to improve the coding quality. This approach has the advantage of taking into account by construction the effects of the texture and contour coding techniques in the segmentation process.

As demonstrated by several examples, this segmentation-based coding approach is robust and simple. It makes no assumption about the input sequences. It produces several coded results from the simplest to the most complex one while preserving the contour information. It is therefore well-suited for progressive coding and transmission applications. The various examples shown demonstrate the interest of the approach for very low bit rate video coding. Finally, let us mention, that the approach presented here works on the basis of a block of frames, and this may present some drawbacks for interactive applications. However, the approach can be extended to a time-recursive version where the concept of block of frames is replaced by a sliding window of frames. This is the topic of our current investigation.

ACKNOWLEDGMENT

The authors are very grateful to Mr. C. Gu for having processed and coded the contour images. This work has been achieved within the framework of the MORPHECO project of the RACE program of the European Community.

REFERENCES

- [1] E. J. Delp and O. R. Mitchell, "Image compression using block truncation coding," *IEEE Transactions Commun.*, vol. COM-27, no. 9, pp. 1335-1341, Sept. 1979.
- [2] M. Eden and M. Kocher, "On the performance of contour coding algorithm in the context of image coding: Part I: Contour coding," in *Signal Processing*, vol. 8, pp. 381-386, 1985.
- [3] C. Gu and M. Kunt, "Contour image sequence coding by motion compensation and morphological filtering," in *Proc. Int. Workshop Coding Techniques Very Low Bit-Rate Video* (Colchester, UK), Apr. 7-8, 1994.
- [4] A. K. Jain, "Image data compression: A review," *Proc. IEEE*, vol. 69, pp. 349-389, 1981.
- [5] M. Kocher and M. Kunt, "A contour-texture approach to picture coding," in *Proc. 1982 IEEE Int. Conf. Acoust. Speech Signal Processing* (Paris, France), May 1982, pp. 436-440.
- [6] M. Kunt, A. Ikonomopoulos, and M. Kocher, "Second generation image-coding techniques," *Proc. IEEE*, vol. 73, pp. 549-574, Apr. 1985.

- [7] F. Marqués, J. Sauleda, and A. Gasull, "Shape and location coding for contour images," in *Proc. Picture Coding Symp. PCS'93* (Lausanne, Switzerland), Mar. 1993, pp. 18.6.1–18.6.2.
- [8] F. Meyer and S. Beucher, "Morphological segmentation," *J. Visual Commun. Image Representat.*, vol. 1, no. 1, pp. 21–46, Sept. 1990.
- [9] F. Meyer, "Color image segmentation," in *Proc. Int. Conf. Image Processing* (Maastricht, The Netherlands), 1992.
- [10] M. Pardàs, P. Salembier, and L. Torres, "3D Morphological segmentation for image sequence processing," in *Proc. IEEE Workshop Nonlinear Signal Processing* (Tampere, Finland), 1993, pp. 6.1.3.1–6.1.3.6.
- [11] S. Rajala, M. Civanlar, and W. Lee, "Video data compression using three-dimensional segmentation based on HVS properties," in *Proc. of IEEE Int. Conf. Acoust. Speech Signal Processing'88* (New York, NY), 1988, pp. 1092–1095.
- [12] P. Salembier and M. Kunt, "Size-sensitive multiresolution decomposition of images with rank order based filters," in *Signal Processing*, vol. 27, no. 2, pp. 205–241, May 1992.
- [13] P. Salembier and J. Serra, "Morphological multiscale segmentation of images," in *Proc. SPIE Visual Commun. Image Processing'92* (Boston, MA), 1992, pp. 620–631.
- [14] P. Salembier, J. Serra, and J. Bangham, "Edge versus contrast estimation of morphological filters," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Processing '93*, (Minneapolis, MN), 1993, pp. V.45–V.48.
- [15] P. Salembier, "Morphological multiscale segmentation for image coding," *Signal Processing*, vol. 38, no. 3, Special Issue on Nonlinear Signal Processing, pp. 359–386, Sept. 1990.
- [16] J. Serra, *Image Analysis and Mathematical Morphology*. New York: Academic, 1982.
- [17] ———, *Image Analysis and Mathematical Morphology-Volume II: Theoretical Advances*. London: Academic, 1988.
- [18] L. Vincent, "Algorithmes morphologiques a base de files d'attente et de lacets. Extension aux graphes," Ph.D. Thesis, Paris School of Mines, May 1990.
- [19] L. Vincent and P. Soille, "Watersheds in digital spaces: An efficient algorithm based on immersion simulations," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 13, no. 6, pp. 583–598, June 1991.
- [20] P. Willemin, T. Reed, and M. Kunt, "Image sequences coding by split and merge," *IEEE Trans. Commun.*, vol. 39, no. 12, pp. 1845–1855, Dec. 1991.



Philippe Salembier graduated from Ecole Polytechnique, Paris, France, in 1983 and from Ecole Nationale Supérieure des Télécommunications, Paris, France, in 1985. He received the Ph.D. degree from the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, in June of 1991.

From September 1985 to March 1989, he worked at Laboratoire d'Electronique Philips at Limeil-Brevannes, France, in the field of communications systems applied to television signals. In April 1989, he joined EPFL to work on image processing. From July 1991 to November 1991, he was with Harvard Robotics Laboratory, Cambridge, MA. In November 1991, he joined the Polytechnic University of Catalunya, Barcelona, Spain, where he is teaching digital signal processing. His current research interests include image and sequence compression, image analysis, nonlinear signal processing, and mathematical morphology.



Montse Pardàs received the M.S. degree in telecommunications in 1991 from the Polytechnic University of Catalunya, Barcelona, Spain. She is currently working on her Ph.D. thesis in the Signal Processing Department.

Her main research activity deals with nonlinear signal processing with a special emphasis on mathematical morphology. She is particularly interested in analysis problems for sequences such as segmentation, motion estimation and coding.