MOTION COMPENSATED PARTITION CODING

Philippe Salembier

Dept. of Signal Theory and Communications
Universitat Politècnica de Catalunya
Campus Nord - Módulo D5
C/ Gran Capità, 08034 Barcelona, Spain
Tel: (343) 401 74 04, Fax: (343) 401 64 47
E-mail: philippe@gps.tsc.upc.es

ABSTRACT

This paper deals with the coding of the partition information resulting from the segmentation of video sequences. Motion compensation of partition sequences is described as an efficient inter-frame mode of coding. It involves the prediction of the partition, the computation of the partition compensation error, the simplification of the error and its transmission. The major issues and processing steps of a general motion compensation loop for partitions are presented and discussed.

KEYWORDS: Segmentation-based video coding, Very low bit rate, Partition, Motion compensated, Region tracking.

1 INTRODUCTION

Segmentation-based coding of video sequences is rapidly becoming a very active field of research.[10,6,9,5] Its basic idea is to define a partition of the image sequence that is more appropriate for coding than the block partition used in more classical coding schemes. The coding algorithms involve at least three steps: 1) the definition of the partition, 2) the coding and transmission of the partition and 3) the coding and transmission of the pixel values inside each region defined by the partition. Several segmentation criteria can be used to define the partition. Let us mention in particular the homogeneity in gray level value,[10,9] the classification in static or moving region,[6] the homogeneity in motion,[1] etc. In all cases, the resulting partition depends on the input video sequence and has to be transmitted to the receiver. This transmission is the objective of the so-called *partition coding*.

The partition sequence is a sequence defining the shape of the regions and their time evolution. The most simple way to define a partition sequence is to assign to each pixel of the sequence a specific number, called a *label*, defining the region it belongs to. In the following we will assume that the segmentation has solved the region correspondence problem, see[9,5] for example. As a result, a region can be tracked in the successive frames. The information contained in a partition sequence is composed of the shape of each region, their position within each frame and their label. Note that the problem is therefore more complex than the pure *contour coding* problem which deals only with coding of the shape of the regions. In the sequel, the term *partition coding* will refer to the coding of the shape, position and labels.

As in any video coding scheme, two modes of transmission can be distinguished: *intra-frame* and *inter-frame*. The intra-frame mode consists of sending the information about the partition of each frame independently. This mode has to be used for example for still images, for the first frame of a sequence or even periodically in order to totally refresh the information in the receiver. By contrast, the inter-frame mode relies on the characterization and coding of the time evolution of the partition from one frame to the next one. This approach is generally implemented via motion compensation. The objective of this paper is to describe a general motion compensation strategy for partitions. We will present and discuss the main issues of this coding strategy, in particular, the type of motion information to be used, the problem of overlapping regions and the structure of the compensation loop. Our objective is to describe a general scheme that does not rely on a specific contour representation and does not imply a specific segmentation scheme.

The organization of this paper is as follows: the following section describes the problem of motion compensation of partition sequence. Section 3 is devoted to the estimation of the order and transmission mode that are necessary for motion compensation. Section 4 describes with some details the coding process and results are reported in section 5.

2 MOTION COMPENSATION OF PARTITIONS

As illustrated in Figure 1, the inter-frame mode of partition coding relies on the estimation of the motion of the partition between two successive frames, the prediction by motion compensation, the computation of the prediction error, the simplification of this error and the coding of the simplified error. This scheme is similar to the one classically used for motion compensation of texture, but it is applied here on frames of label.

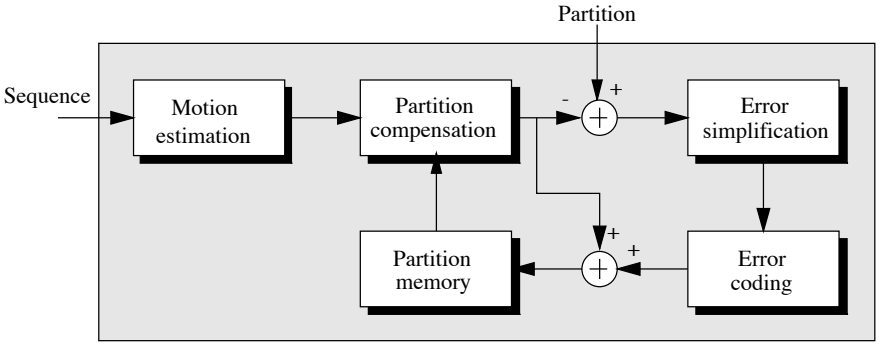


Figure 1: Motion compensated partition coding

The problem of partition compensation is illustrated in Figure 2. Before compensation, the motion of the various regions of the current partition *seg* has been estimated (Figure 2.A). The estimation gives a set of parameters describing the time evolution of each region in a backward mode. The first question to answer is to define which motion can be used to compensate the partition. Note that, texture motion and shape motion may not be equivalent. Both motions coincide in the case of a foreground region. Indeed, for such a region, the pixels of its interior follow the same motion as its contours. This is, however, not the case for a background region because the modifications of its shape or of its contours are defined by the motion of the regions in its foreground. Following the discussion of,[7] we will assume that the texture motion is used to compensate both the partition and the texture. The partition compensation problem is shown in Figure 2.B: based on the previously coded partition $rec(T-1)$, that has been stored in the receiver, and on the motion parameters, the compensation should define a prediction of the current partition $rec(T)$.

The compensation itself can work either in a *forward* mode or in a *backward* mode. In the forward mode (see in Figure 3), the pixels of $rec(T-1)$ are projected towards $rec(T)$. This projection can be done if the motion

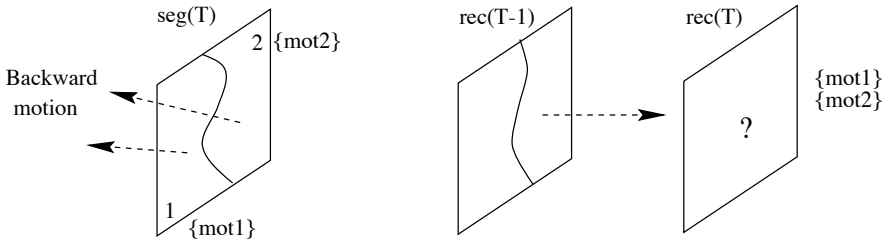


Figure 2: Partition compensation

vectors, as defined by the estimation done in a backward mode, are inverted. As can be seen, two problems may result from the transformation of the regions. Some pixels of $rec(T)$ may have no label, they constitute the so-called *empty areas*. By contrast, some pixels may have several label candidates, these conflicting areas are called *overlapping areas*. To solve the conflicts, an extra information called the *order* can be used.[7] The order information is used to decide which region is considered to be in the foreground of which region. In case of conflicts between labels, the foreground region gives the correct label. The use of the texture motion and of the order is a quite efficient solution because the texture motion information leads to a good compensation of the texture and the order only represents a small amount of information. Finally, the empty areas are left without label and are processed as compensation errors.

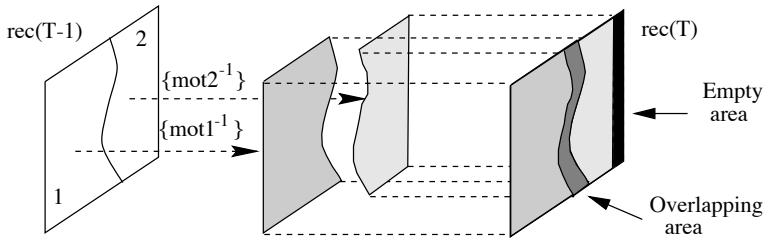


Figure 3: Forward motion compensation of partitions

The dual mode of compensation is illustrated in Figure 4. It is a backward mode in the sense that, for each pixel of $rec(T)$, one tries to look on $rec(T-1)$, which label has to be selected. In this case, the main problem is to define which motion vector has to be used when the pixel (i,j) of $rec(T)$ is considered. Indeed, since the current partition is not yet known on the receiver side, one does not know the region the pixel belongs to and therefore its corresponding motion model. The solution consists in considering all possible vectors defined by all possible regions. In the case of Figure 4, there are two regions, therefore, two vectors are considered for each point: one given by region 1 $\{mot1\}$ and one given by region 2 $\{mot2\}$. Each time a vector as defined by region n does not point to a pixel belonging to region n in $rec(T-1)$, the compensation is considered as being invalid and is discarded. In Figure 4, this is the case for two vectors. As in the case of forward motion compensation, some pixels have no valid compensation (*empty areas*) and some others have more than one candidate (*overlapping areas*). As previously, the order information is used to solve the conflicting areas.

The main difference between the forward and backward modes of compensation deals with the quantization of the pixel locations. Indeed, generally, the motion vectors start from an integer pixel location but point to a non-integer location. In the forward case, it is the locations of pixels of $rec(T)$ that have to be quantized whereas in the backward case, the locations of pixels of $rec(T-1)$ have to be quantized. There are some difficulties related to the forward mode in the case of motion models involving modifications of the scale (zoom in particular). Indeed, in the case of region expansion, the modification of the distance between two pixels may create more empty areas

in the compensated frame. These problems can be solved[7] but, in general, the backward mode is more simple.

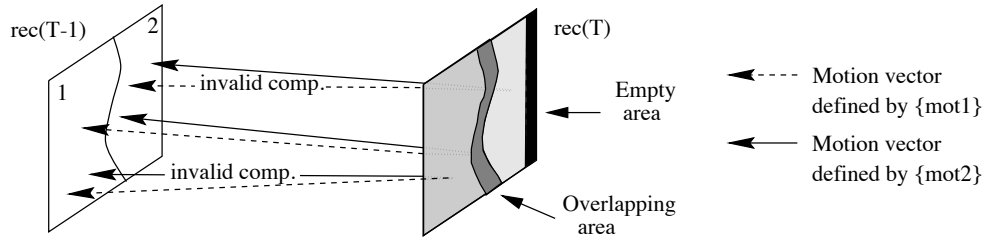


Figure 4: Backward motion compensation of partitions

Two examples of partition coding using forward motion compensation can be found in.[2,3] The main difference between these two works consists in the coding of the error which is done by the coding of its contours (chain code) in[3] or of its shape (Geodesic skeleton) in.[2] Both techniques rely on the estimation of the motion of each region and the processing of regions individually. That is, the motion of the shape of each region is estimated (not the texture motion), then each region is compensated and finally, the error resulting from the region compensation is coded and transmitted individually for each region.

In,[6] the texture motion is used to compensate a mask defining the moving area and no order information is necessary because the model assumes *a priori* that there are only one moving foreground region (the mask to compensate) and one static background region. Note that the motion compensation only deals with a binary mask. This situation can be viewed as a simplified version of the general problem that is discussed here: we want to deal with complex partitions, each region of which can have its motion, and various foreground/background relations with respect to other regions.

Let us start by the description of the order estimation. In the sequel, we will assume the following notations:

- *seg* represents the current (at time T) partition.
- *rec* denotes the previous (at time $T-1$) partition reconstructed on the receiver side.
- *motion* and *order* are the motion and order information that characterize the evolution of the partition between time $T-1$ and T.
- *original* is the original frame at time T.

3 ORDER AND TRANSMISSION MODE ESTIMATION

The goal of the order estimation is of course to estimate the order for the partition compensation. In practice, this order is used on the receiver side to solve the conflicts between compensated regions. Conflicts appear on locations where several region labels overlap. This notion is implicitly used in[3,9] where a transmission order is defined. A transmitted region is supposed to be in the background of all previously transmitted regions. The transmission order is defined by estimating the prediction error for each region individually and the region creating the lowest error is transmitted first. This solution is not optimum because it assigns one order to each region and does not try to solve the possible conflicts between pairs of regions. For example, it cannot deal with situations such as an object A being in the foreground of an object B which is in the foreground of C which, itself, has some parts in the foreground of A. Depending on the segmentation criteria, such situations may happen for scenes with 3D objects and several depth levels. So, the real objective of the order estimation is to solve all the possible conflicts between pairs of regions.

One may think that the problem of overlapping regions may be solved by choosing a specific strategy in the receiver side and by processing the resulting errors as compensation errors. From our experience, this is not true because, first, the order information only represents a small overhead, and second, the motion information that is used does not characterize the shape motion. Finally, let us mention that the order information seems to be of vital importance for motion models that are more complex than the simple translation.

Moreover, in practice, it may not be always efficient to send all regions in inter-frame mode (that is by compensation). For example, if the motion estimation has not produced reliable results or if the motion of the region cannot be modeled by the motion estimation, then the prediction error of the region shape may be so large that it is less expensive to directly code the entire region. In this case one has to switch from inter-frame mode to intra-frame mode. This idea leads to the definition of a transmission mode (intra or inter) for each region. This solution is in contrast to the work reported in[3,2] where all regions of the partition are motion compensated. As a result, the gray level and color values of one region can be sent in inter-frame mode and its shape and location can be sent either in intra-frame or in inter-frame.

The 'Order computation' and the 'Mode estimation' are interdependent. Indeed, the 'Order computation' needs to know the transmission mode of each region and the 'Mode estimation' needs to know how to compensate each region. That is, it needs to know the output of the 'Order estimation'. This leads to the decision-directed loop shown in Figure 5. The input data are the current partition *seg*, the previous reconstructed partition *rec* as well as the motion information *motion*. To start the process, all new regions (that is, regions present in *seg* but not present in *rec*) are assumed to be transmitted in intra. Based on this assumption, the order is estimated. Then, assuming that the estimated order is correct, the transmission mode is estimated and some transmission modes may be modified. If this is the case, the order has to be estimated again, and so on. In practice, a few iterations (typically two or three) are performed. The results are the order and a list of transmission modes. In the following we describe more precisely the two blocks of Figure 5.

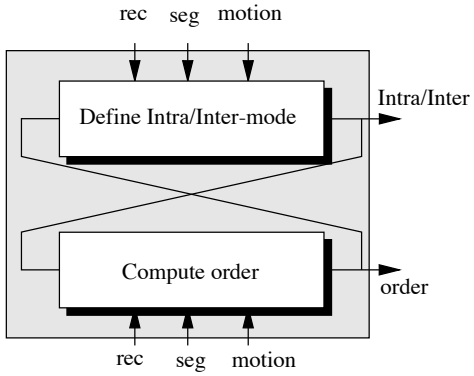


Figure 5: Decision-directed loop for the order estimation: order computation and intra/inter-mode estimation

3.1 Define intra/inter-mode

The objective of this processing step is to define the transmission mode (intra or inter) of each region. This decision is taken on the basis of the cost of the contour. Note that, one region can be sent in intra-frame mode for partition coding and in inter-frame mode for texture coding. To deal with this situation, one has to add a binary information to say to the receiver whether or not the region is transmitted in intra-frame mode for partition coding. This information can be stored with the motion information (*motion* will therefore involve a set of motion parameters for each region plus a binary information indicating the transmission mode of the contours).

The decision on the transmission mode is simply achieved by comparing the cost of the region shape if it is sent in intra-frame mode or in inter-frame mode (that is by motion compensation). Based on the previous partition *rec*,

the order and motion information, the whole partition is compensated and the compensation error is computed. Finally, the transmission mode selection is done by comparing for each region the cost of transmission of the region as defined by the segmentation (intra-frame mode) or as defined by the error (inter-frame mode). Note that the transmission cost can assessed either by actually coding the region or by simply making an estimation.

3.2 Compute order

The order computation can be decomposed in two steps as shown in Figure 6. The first step ('Estimate overlapping') estimates the conflicts between regions during the compensation and the second one ('Define order') achieves a quantization of the order information for the transmission. In the following we will assume a backward mode of compensation, but the principles remain the same for a forward mode.

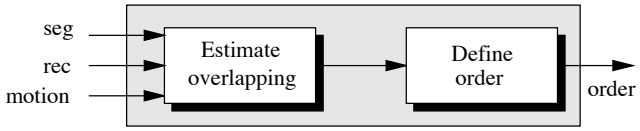


Figure 6: Description of the 'Compute order'

The 'Estimate overlapping' performs a backward motion compensation of the partition as described in section 2. Of course, only valid compensations are taken into account. The overlap information consists of a list of possible regions in conflict. At each location where there is a conflict between two labels, the corresponding list entry is updated indicating which label is in the foreground. The foreground label is defined by the label that is actually present at that location in the segmented frame *seg* (see[8] for more details).

At the end, the overlap list gives the number of occurrences where it has been possible to declare that a given region is in the foreground of another region. Finally, the order has to be quantized because the receiver only needs a binary decision in order to be able to resolve the conflicting labels during the compensation. The quantization is achieved by examining each pair of labels and by comparing the number of occurrences where the first one has been declared to be in the foreground of the second one with the number of occurrences where the second one has been declared to be in the foreground of the first one. The final order between the two regions is defined as the one corresponding to the largest number of occurrences.

Finally, *order* is entropy coded and sent through the transmission channel. The order, being a binary information, can be very efficiently coded (see Section 5)

4 INTER-FRAME PARTITION CODING

Once the order and the transmission mode have been estimated, the actual coding of the partition can be done. The structure of the encoding process is illustrated by Figure 7. First, all regions which have to be sent in intra-frame mode are processed. As explained previously, these regions are either new regions (region that are not present in *rec*) or regions that cannot be well compensated and are more efficiently sent in intra-frame mode. The second step consists in computing a partition involving the regions sent in intra-frame mode and the compensated regions. This is the objective of the 'Compensate partition' block. Finally, the partition errors are extracted, simplified and coded by the 'Code inter-regions' block. In the following, we describe more precisely these three steps.

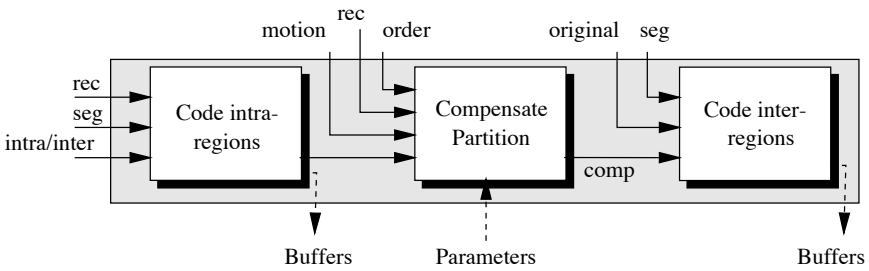


Figure 7: Structure of the inter-frame partition coding

4.1 Code intra-regions

The objective of this processing step is twofold: first, to send the contour information of the regions transmitted in intra-frame mode and second to create a binary mask defining where the partition is considered as being already defined. This mask will be useful during the compensation because each time a label is compensated at a location corresponding to a region transmitted in intra-frame mode then the compensated label will not be taken into account.

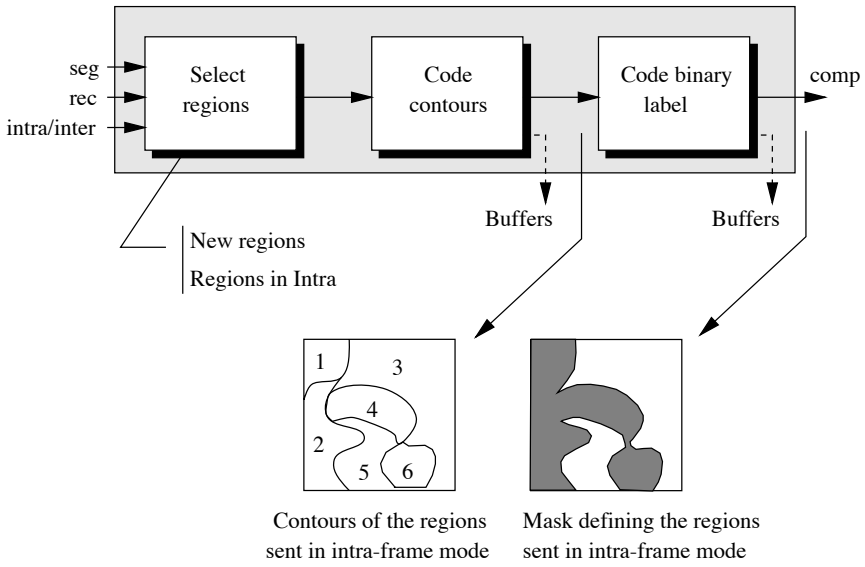


Figure 8: Description of the 'Code intra-regions' block

The coding in intra is performed in three steps as described in Figure 8. The first step consists in extracting the set of regions which have to be sent in intra. The 'Select regions' block defines a partition of the space which has to be sent to the receiver. Almost all contour coding techniques (lossy or lossless) may be used (a review can be found in[8]). As can be seen in Figure 8, the information which is transmitted by the 'Code contours' block allows the restoration on the receiver side of the contours of the intra-regions. However, it does not say which region is transmitted in intra. For example, in Figure 8, one can see that the transmission of the contour information creates a partition of 6 regions. Assume, for example, that from these 6 regions, only 4 are sent in intra-frame mode: region number 1, 2, 4 and 6. In this case, we want to create a mask indicating the locations where the label has been sent in intra-frame mode. This is the purpose of the 'Code binary label'. It sends different codes for each region depending on its transmission mode. In the following, the mask information is assumed to be stored in an image called *comp*, that is the first version of the compensated partition.

4.2 Compensate partition

The compensation of the previous partition is similar to the one used for the order estimation (see section 2). The compensated information can be postprocessed in order to create a partition. Indeed, the compensation technique does not guarantee the definition of connected regions and in practice several connected components may have the same label. In some applications, it is desirable to have only one connected component per label. In this case, the two step procedure described in Figure 9 may be used. The first step selects one connected component for each label. Several criteria may be used. A simple one consists of selecting the largest component (note that this selection can be done on the receiver side without the need of transmission of any overhead information). The elimination of some connected component will create some 'holes' in the partition, that is regions which correspond neither to a region sent in intra-frame mode nor to a compensated label. These holes can be left and processed as individual regions by the following step or they can be removed by the 'Resolve partition holes' block (see Figure 9). Here also several techniques may be used, in particular geometrical techniques are attractive because they can be implemented in the receiver with no transmission cost. For example, the hole can be eliminated by a propagation of the neighboring labels or by assigning it to the largest neighboring region, etc.

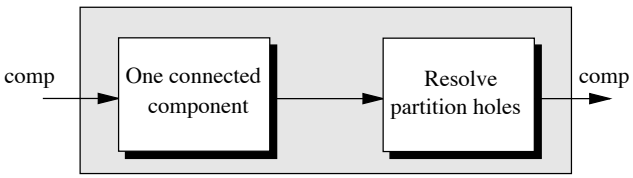


Figure 9: Postprocessing of the compensated partition

At the end of this procedure, one has the final compensated image (stored in *comp*). The final step of the encoding process deals with the compensation errors.

4.3 Code inter-regions

Figure 10.A describes the organization of the partition coding in inter-frame mode. It consists of computing the error of the compensated partition, in simplifying part of the error, and in sending the information about the contours and labels of the error.

- *Compute error partition:* The error computation consists in extracting the locations of the space where the compensated label *comp* is different from the current label defined by *seg*. This results in an error mask which has to be labeled. The error mask is labeled in such a way that in each resulting region of the error, both the labels of the compensated partition *comp* and of the current segmentation *seg* be constant values. Figure 10.B illustrates a simple example where the segmentation is made of three regions and the compensated partition only involves two regions. The error partition is also shown and three regions of error have been created with label a, b and c.
- *Simplify error partition:* This step introduces the major part of the losses in the the coding process. Each region of the error is examined to know whether it has to be preserved and coded or discarded. In practice, the most useful criteria are:
  1. Geometrical criterion: All error regions that are smaller than a given size may be removed.[3,2,6,9] Note that the size of the region can be considered as its area or its maximum elongation or its maximum width, etc.
  2. Gray level criterion: An error region can be discarded if it will not introduce a strong modification of the gray level (or color) value. The decision can be taken by considering the values assigned to the

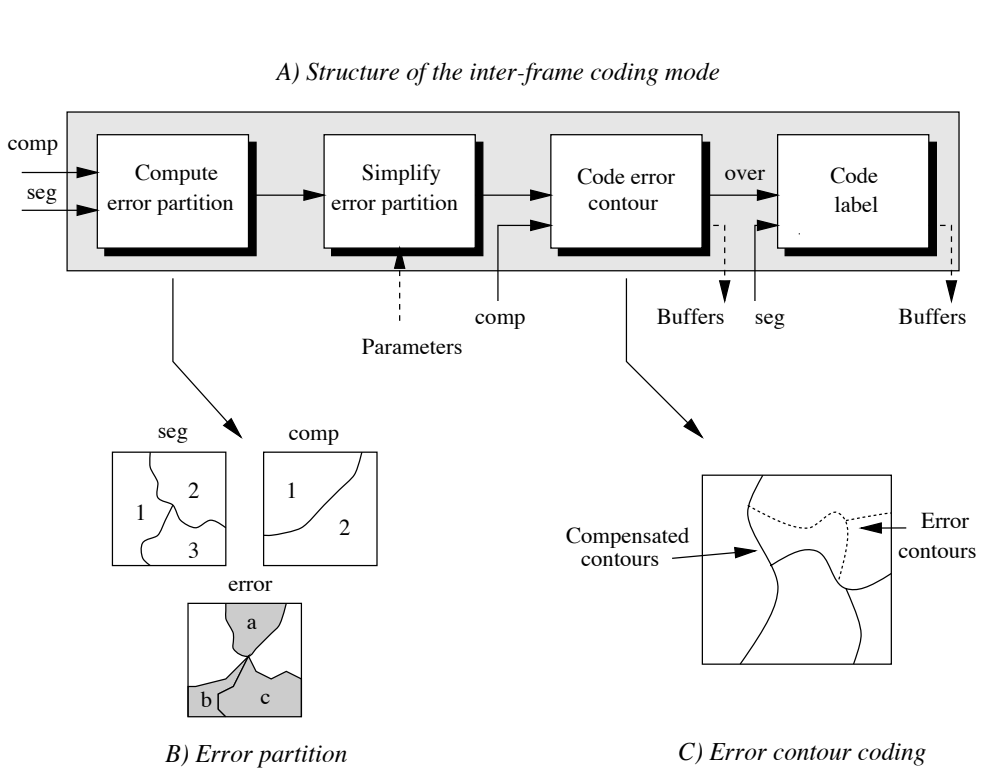


Figure 10: Description of the 'Code inter-regions'

error region if it is assumed to belong to the region defined by *comp*, on the one hand side, and by *seg*, on the other hand side. If the difference is small, the error region can be discarded.

- *Code error contour:* The actual coding of the error is done in two steps. The first step consists in coding the contours of the error regions. This coding step is presented in Figure 10.C. It relies on a differential coding of the contours. Indeed, the receiver already knows the contours that correspond to the compensated partition *comp*. Therefore, only, the new contours (dotted lines) have to be sent. Techniques similar to the one used for the intra-frame mode ('Code partition intra' of Figure 7) can be used. The only difference is that one has to take into account the presence of already known contours. In particular, the starting points of the new contours are always located on already known contours. This information can be used to efficiently code these starting points. Moreover, new contours end when they reach a contour of the compensated partition. Classical examples can be found in.[3,2] The resulting partition *over* can be seen as an 'over-partition' because it involves contours of the compensated partition and of the segmentation.

- *Code label:* The final coding step is to send the label of each region of the over-partition *over*. As commented above, the label information has to be restored on the receiver side in order to be able to define and to track the time evolution of the various regions. First, the labels assigned on the receiver side are extracted. These labels, called *compensated_label*, are the labels which were defined by the compensation process. Second, the most probable labels of the segmentation *seg* are estimated. Note that, because of the simplification step, each region of the over-partition *over* may involve several labels of the segmentation *seg* (this is not the case for the compensated partition *comp*). The most probable label is defined as the dominant label (label with the highest number of pixels) of *seg* that falls within the over-partition region, say *dominant_label*.

  The labels can be sent directly in the transmission channel but, in general, this will result in an excessive amount of information. To reduce this amount of information one can use the following coding strategy: For a given region, consider the set of compensated labels of its neighboring regions. Denote by {*compensated_neigh_label*} this set of labels and create an ordered list of these labels. The order can

be defined simply by the numerical values of the labels or by the geometrical position of the neighboring regions. Finally, one can assign the following code to define the label:

- If *dominant_label* is equal to *compensated_label*, then, assign the label code 0.
- If *dominant_label* is equal to one of the labels in the ordered list $\{compensated\_neigh\_label\}_n$, assign a label code indicating the position of the label in the order list.
- In all other cases, send an escape symbol followed by the actual label *dominant_label*.

This strategy is in general quite efficient because, most of the time, the *dominant_label* is either the *compensated_label* or a label of the neighboring regions. Moreover, the number of the neighboring regions is rather small (typically 4 or 5) which means that the entropy of the symbol (position of the label in the ordered list) is small. These symbols are finally entropy coded and sent.

5 RESULTS

The purpose of this section is to describe the behavior of the motion compensated partition coding technique that has been described previously. Note that the performance of the method depends on two important points: the quality of the motion estimation and the kind of contour representation. It is not our purpose here to describe a particular motion estimation technique. However, in order to code the partition, we have used here a motion estimation technique that assumes an affine model and estimates a set of six parameters for each region. Moreover, the general strategy of motion compensation of partition can be used with almost any kind of contour representation. For the practical coding of the partition, we have used a chain code representation. The results given in this section will therefore be related to this representation, however, most of the behavior of the approach (not the actual figures) can be translated to other representations.

As a reference, let us consider the case of intra-frame transmission mode. Table 1 describes the performances of a modified chain code technique[4] to code two partition sequences: *Foreman* and *News*. The figures are given in *bits per contour points (bpcp)*. As can be seen, in both cases, most of the information is devoted to the coding of the individual movements of the chain. In the case of the *News* sequence, the results are better than for the *Foreman* sequence. This is due to the complexity of the contours that are simpler in the first case. In fact, the segmentation of *News* involves a large number of straight lines, this increases the probability of one movement and the entropy coding takes advantage of this higher probability.

Sequence	Chain moves	Starting points	Labels	Total
Foreman	1.25	0.01	0.05	1.31
News	1.08	0.02	0.03	1.13

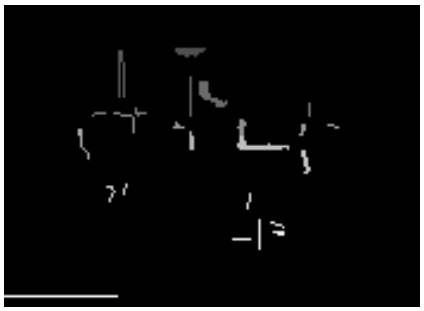
Table 1: Coding result in the intra-frame mode (figures are in bits per contour points)

Figure 11 illustrates the whole process of partition coding by motion compensation. First, the intra-frame regions are transmitted (Figure 11.A). Second, the previously coded partition is motion compensated (Figure 11.B). In case of conflict between two labels, the order information is used to take a decision. Figure 11.C shows the compensation error after simplification. The simplification has removed the components of size lower than five pixels as well as the components that will not produce a gray level error higher than five. The contours of the error are sent to the receiver to create the over-partition. Finally, the coded partition can be seen in Figure 11.D. Coding figures about this sequence can be found in Table 5. As can be seen, the global cost has been severely reduced.

Finally, Table 5 illustrates the influence of two error simplification criteria (sequence *Mother and Daughter*).

A) Regions transmitted in intra-frame mode

B) Compensated partition (pixels with no label are in black)



C) Partition error

D) Final partition

Figure 11: Partition coding steps

Sequence	Intra-frame	Order	Chain moves in inter	Starting points	Label	Total
News	0.26	0.12	0.14	0.10	0.10	0.72

Table 2: Coding result in the inter-frame mode (figures are in bits per contour points)

Important reduction on the number of bits per contour point can be obtained. Of course, the definition of a proper simplification value should depend on the final application.

6 CONCLUSIONS

Coding of partition sequences is one of the major issues of second generation video coding. Creating a signal-dependent partition suitable for efficient coding or for the so-called object-oriented functionalities is a viable solution only if its resulting transmission cost is not prohibitive. It is very likely that the major issue of partition sequence coding concerns the inter-frame mode. This is a rather new field of research and few contributions have been made. In this paper, we have tried to deal with the major issues of motion compensation of partition sequences. It is believed that motion compensation is a good solution to take benefit of the temporal correlation that exists between frames in a partition sequence. As a consequence, we have described the various steps of a general motion compensation loop. Each step has been discussed in a formal way without relying on a specific

Size	0	5 pixels	10 pixels
bpcp	0.88	0.46	0.32
Gray level	0	10	20
bpcp	0.88	0.82	0.64

Table 3: Simplification of the partition error

contour representation or a specific partition. Current results have shown that the use of compensation can divide at least by a factor of two the partition coding cost with respect to the intra-frame mode.

7 ACKNOWLEDGMENTS

This work has been supported by the *Morpheco* and *Mavt* projects of the European RACE Program.

8 REFERENCES

[1] P. Bouthemy and E. François. Motion segmentation and qualitative dynamic scene analysis from an image sequence. *International Journal of Computer Vision*, 10(2):157–182, 1993.

[2] P. Brigger and M. Kunt. Contour image sequence coding using the geodesic morphological skeleton. In *International Workshop on Coding Techniques for Very Low Bit-rate Video*, pages 3.1–3.2, Essex, Colchester, April 1994.

[3] C. Gu and M. Kunt. 3D contour image sequence coding based on morphological filters and motion compensation. In *International Workshop on Coding Techniques for Very Low Bit-rate Video*, Colchester, U.K., April 1994.

[4] F. Marqués, J. Sauleda, and A. Gasull. Shape and location coding for contour images. In *Picture Coding Symposium*, pages 18.6.1–18.6.2, Lausanne, Switzerland, March 1993.

[5] F. Marqués, V. Vera, and A. Gasull. A hierarchical image sequence model for segmentation: Application to object-based sequence coding. In *Proc. SPIE Visual Communication and Signal Processing-94 Conference*, pages 554–563, Chicago, USA, Oct 1994.

[6] H.G. Musmann, M. Hőtter, and J. Ostermann. Object-oriented analysis-synthesis coding of moving images. *Signal Processing, Image Communication*, 1(2):117–138, October 1989.

[7] M. Pardàs, P. Salembier, and B. González. Motion and region overlapping estimation for segmentation-based video coding. In *IEEE International Conference on Image Processing, ICIP'94*, volume II, pages 428–432, Austin, Texas, November 1994.

[8] P. Salembier, F. Marqués, and A. Gasull. Coding of partition sequences. In L. Torres and M. Kunt, editors, *Video Coding: The Second Generation Approach*. Kluwer, 1996. ISBN: 0 7923 9680 4.

[9] P. Salembier, L. Torres, F. Meyer, and C. Gu. Region-based video coding using mathematical morphology. *Proceedings of IEEE (Invited paper)*, 83(6):843–857, June 1995.

[10] P. Willemin, T. Reed, and M. Kunt. Image sequence coding by split and merge. *IEEE Transactions on Communications*, 39(12):1845–1855, December 1991.