Dynamic Approach to Visual Data Compression

E. Reusens [*], T. Ebrahimi, C. Le Buhan, R. Castagno
V. Vaerman, L. Piron, C. de Solà Fàbregas, S. Bhattacharjee, F. Bossen
and M. Kunt
Signal Processing Laboratory
Swiss Federal Institute of Technology at Lausanne - EPFL
1015 Lausanne
Switzerland

Abstract

This paper presents the *EPFL* proposal to MPEG-4 video coding standardization activity [1]. The proposed technique is based on a novel approach to audio-visual data compression entitled *dynamic coding*. The newly born multimedia environment supports a plethora of applications which can not be covered adequately by a single compression technique. Dynamic coding offers the opportunity to combine several compression techniques and segmentation strategies. Given a particular application, these two degrees of freedom can be constrained and assembled in order to produce a particular profile which meets the set of specifications dictated by the application.

The basic principles of this approach are presented together with the data representation system. The major characteristics of dynamic coding are reviewed, along with simulation results showing the performance of such an approach in a very low bitrate video coding environment.

1 Introduction

Early works addressing audio-visual data representation concentrated on specific and well-defined single applications. Among others, one can mention the recommendation ITU-T H.261 for tele-conferencing and video-telephony [2, 3], MPEG-1 for storage on CD-ROM [4, 5], MPEG-2 for higher quality generic coding [6], and the recommendation ITU-T H.263 for very low bitrate coding [7]. These techniques exhibit

[*] Emmanuel Reusens is now with Logitech Inc., 6505 Kaiser Drive, Fremont, CA 94555

outstanding performance when demonstrated within the framework of their respective applications. This is partially due to considerable efforts that have been devoted for several years to the tuning of these systems up to their performance limits.

The scope of audio-visual data representation has changed radically with the birth of multimedia applications. Multimedia is a platform for exchanging information coming from different sources of possibly different nature. On this account, it will support a very broad spectrum of applications. Due to their inherent rigidity, current standards can not adequately address the new expectations and requirements that arisen from such a diversity of applications.

The variety of applications makes the audio-visual data representation problem a challenging issue. The range of applications include tele-medicine, image data retrieval, video-telephony and video messaging, remote monitoring, television, video-on-demand, to name a few. Each application dictates a set of specifications which may greatly vary from one application to another. Diversity of applications means diversity of collection of specifications. Each application is characterized by: the type of data to be processed (still pictures, video, stereo images, etc.), the nature of visual data (natural, synthetic, text, medical, graphics, etc.), the targeted bitrate (low, medium, high), the maximum admissible delay (ranging from real time to off-line), the type of communication (point to point, point to multi-point, multi-point to multi-point), and a set of functionalities (scalability, object manipulation, progressive transmission, editing, etc.).

The major difficulty resides in the fact that, as widely acknowledged, no universal coding technique exists. In other words, no single coding technique will be able to meet the requirements of the entire range of applications. A compression technique that is adequate for a given application may be totally inappropriate for another. For example, a compression technique dedicated to medical imaging applications is likely to be inappropriate for head-and-shoulder video-telephony.

In this context, a straightforward solution would be to design a compression algorithm for each specific application. However, this option is not only technically infeasible, but also promotes a short-sighted approach which would prohibit rapid integration of emerging applications.

This work proposes a unified approach to audio-visual data representation called *dynamic coding*. This denomination stands for the dynamic combination of multiple compression techniques within the same framework. Dynamic coding is not a particular compression algorithm but rather a process enabling an encoder to choose appropriate models for describing portions of given data, themselves obtained according to a pre-defined or automatic procedure [8].

This paper is structured as follows. Section 2 discusses the general dynamic coding approach. The basic principles, the data representation system as well as the procedure of dynamic coding are described. In Sec. 3, a particular implementation of dynamic coding for video compression in the framework of video-telephone/conference applications is detailed. The proposed technique defines a set of admissible coding strategies with respect to the collection of specifications associated to this type of applications. The best solution is obtained by means of a rate-distortion optimization procedure. An extension of the proposed technique enabling object-oriented functionality is also described. Eventually, experimental results of both variants are reported in Sec. 4. Section 5 summarizes the major notions introduced in this paper.

2 Dynamic Coding - General Approach

2.1 Principles

The basic principle of dynamic coding stems from the observation that no coding technique alone is able to properly handle the complete range of applications but rather that each technique is more particularly suited to one particular application. The dynamic coding approach consists of a combination of several compression techniques dynamically activated or discarded according to the environment defined by the application. For instance, as a solution to the diversity of nature of processed data, the proposed approach would allow to select representation techniques suited to the image characteristics. Typically, methods using linear transforms are known to perform well on images with textures, whereas techniques based on fractals perform well on images containing sharp edges and contours [9]. On the other hand, the above-mentioned methods produce poor results on text or graphic images. To illustrate this assertion, Fig. 1 gives samples of typical images that could be processed in a multimedia environment. The three images were compressed using techniques belonging to three major classes of algorithms, namely, a DCT based algorithm (JPEG), a fractal-based technique, and a graphics-oriented method. As assessed by Figs. 2, 3 and 4, none of the three approaches outperforms the others over the three images but rather each of them is more adapted to one type of data.

Dynamic coding is therefore a solution to avoid the weaknesses in a given scheme while maintaining its strong performance when appropriate. The basic idea behind dynamic coding is simple yet very powerful. The data is divided into several regions, each of them encoded using one representation model chosen from a multitude of compression techniques. This approach selects the compression model adaptively

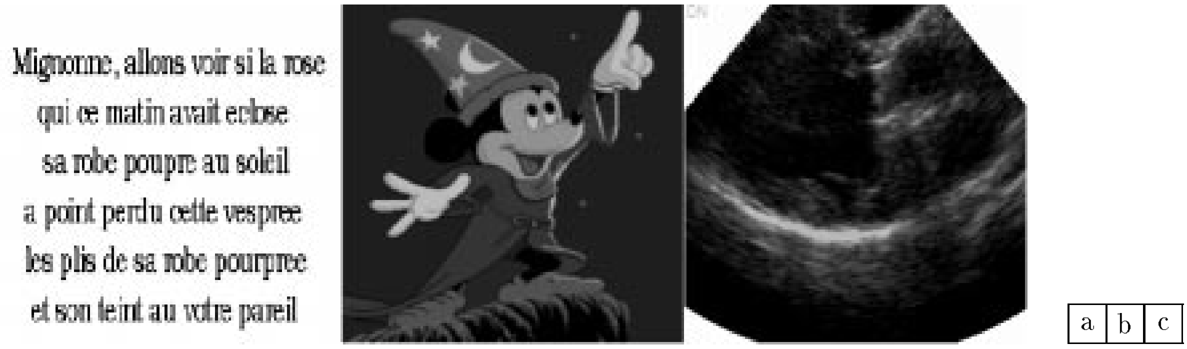
Mignonne, allons voir si la rose
qui ce matin avait eclose
sa robe poupre au soleil
a point perdu cette vespree
les plis de sa robe pourpree
et son teint au votre pareil
a
b
c

Figure 1: Illustration of disparity of data type: a few sample images (a) Poem (b) Fantasia (c) Heart

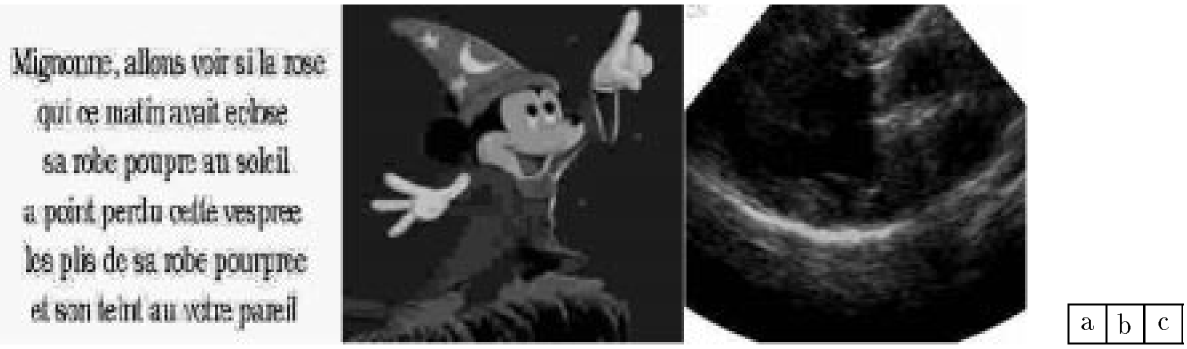
Mignonne, allons voir si la rose
qui ce matin avait eclose
sa robe poupre au soleil
a point perdu cette vespree
les plis de sa robe pourpree
et son teint au votre pareil
a
b
c

Figure 2: JPEG compression: (a) Poem (0.50 bit/pixel) (b) Fantasia (0.25 bit/pixel) (c) Heart (0.40 bit/pixel)

according to the region characteristics. As an example, in areas with texture, a wavelet/subband technique would be used, while in areas containing strong edges and contours morphological methods or other more appropriate techniques will be preferred. Similarly, text areas will trigger an encoding technique which is most appropriate for efficient compression of such data.

2.2 Data Representation System

As discussed above, when compressing a given data by the dynamic coding principle, the data is divided into distinct regions each represented by an appropriate model for an effective compression. Therefore, it becomes clear that dynamic coding is based on two main degrees of freedom, namely, (1) the data segmentation, and (2) the representation model associated to every resulting segment. Figure 5 symbolically illustrates these two degrees of freedom in a graph. The vertical axis corresponds to the data segmentation. Any segmentation algorithm can be applied, ranging from fixed partition into regular blocks up to arbitrary

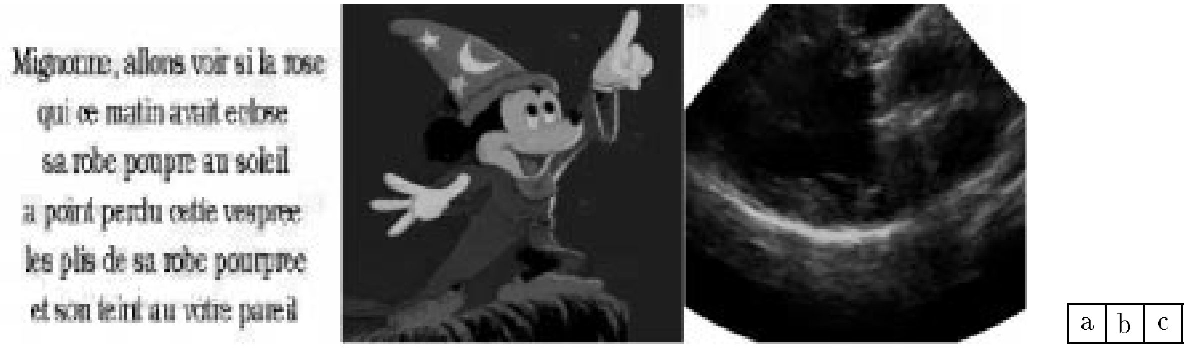
Mignonne, allons voir si la rose
qui ce matin avait eclose
sa robe poupre au soleil
a point perdu cette vespree
les plis de sa robe pourpree
et son teint au votre pareil
a
b
c

Figure 3: Fractal-based compression : (a) Poem (0.50 bit/pixel) (b) Fantasia (0.25 bit/pixel) (c) Heart (0.40 bit/pixel)

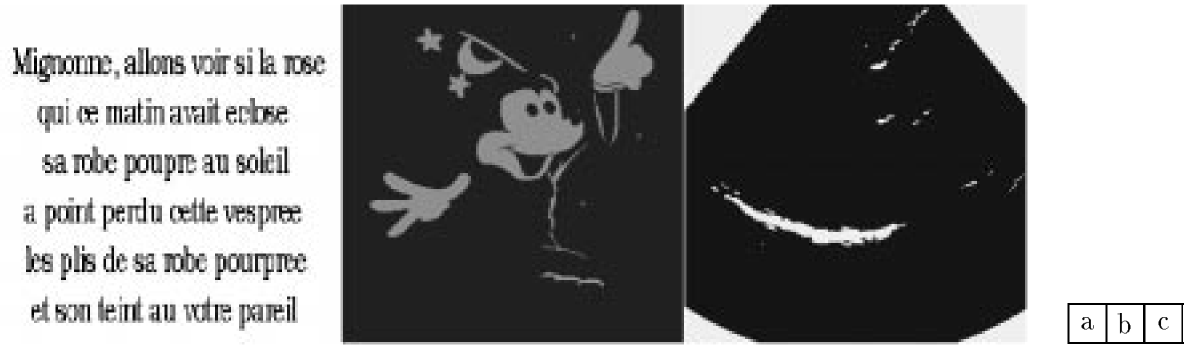
Mignonne, allons voir si la rose
qui ce matin avait eclose
sa robe poupre au soleil
a point perdu cette vespree
les plis de sa robe pourpree
et son teint au votre pareil
a
b
c

Figure 4: Graphic oriented compression (a) Poem (0.50 bit/pixel) (b) Fantasia (0.25 bit/pixel) (c) Heart (0.40 bit/pixel)

shape segmentation through quadtree or polygon based division. The horizontal axis corresponds to the set of coding tools used to describe each region resulting from a given segmentation. These two axes span a plane which reflects all possible coding strategies available within the dynamic coding framework. Each point in the plane represents a very particular coding strategy in which a given scene is segmented into regions of given shape and size, each represented by a given coding technique. Dynamic coding is therefore not a particular compression scheme but rather a process ensuring segmentation and coding tools to interact.

In the context of dynamic coding, a scene description is made up of three components:

- specification of the scene segmentation,
- specification of the representation model associated to each region, (this consists of a flag that informs

the decoder which coding tools have been selected for a given region), and

- the coding parameters associated to each region with respect to the selected representation model.

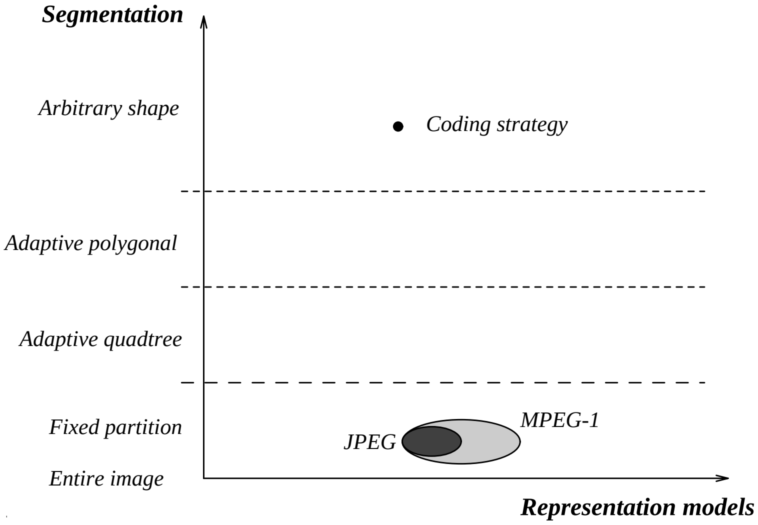

Figure 5: Symbolic description of the data representation system related to the dynamic coding approach

The description of these three components must respect normative requirements in order to specify how a compliant decoder must interpret the compressed bitstream and how the elementary data stream must be processed. For instance, several descriptions of a scene segmentation are possible (chain code, skeleton, tree, polynomial approximation,etc.) and different options may be chosen according to the application. However the option must be specified and each of them must deliver data stream interpretable by the decoder. Similarly, the description produced within a given representation model must follow a syntax known by the decoder. Given the flag specifying the representation model associated to a given region, the decoder uses the appropriate protocol to interpret the data corresponding to the description of the region. The two degrees of freedom allow one a simple and compact description of any specific profile. Compactness is an important property, since the profile specification is, in a certain sense, sent to the decoder. Even if at the decoder side, no matter how the segmentation is performed or how the coding model associated to each region are selected, the bitstream must contain information on the data segmentation and the representation of each resulting regions along with their respective parameters. However the encoder must seek out the best segmentation as well as the most appropriate representation model for every region in the segmented data, taking into account all the constraints imposed by the application. Although dynamic coding relies on two degrees of freedom, most conventional techniques can be seen as particular profiles of a dynamic coding. As an example, a subband/wavelet based technique operating on an entire image can be seen as a particular case of dynamic coding in which the entire image is seen as one single region which is represented by subband/wavelet basis functions corresponding to the filter bank used. Another example would be the JPEG algorithm which can be seen as a particular profile of dynamic coding in which the image is segmented into square regions corresponding to macroblocks and each is encoded using a DCT representation model. The MPEG algorithms can be seen as yet another example which uses a similar segmentation into square regions in which every region is represented by either a DCT representation model or motion compensation models corresponding to coding modes existing in the MPEG approach (see Fig. 5). Thanks to this property, the dynamic coding approach can be made backward compatible with existing standards. On this account the dynamic coding approach can be fully incorporated within the framework of the current verification model (VM) of MPEG-4 [10].

2.3 Dynamic Coding Procedure

As illustrated in Fig. 6, dynamic coding operates along a two-step procedure. The two degrees of freedom (segmentation and representation models) span a space of coding strategies among which many infringe upon the requirements imposed by the application. The first issue is therefore to delimit a set of admissible solutions where the term *admissible* is defined with respect to the application in question. This task is performed by scanning the two degrees of freedom and discarding inappropriate representation models and segmentation strategies. This operation is signal-independent and defines a profile obeying the application specifications by filtering out inadequate coding strategies. For instance, if a particular application requires real-time encoding, representation models and segmentation procedures judged too demanding in terms of computational burden or resulting delays, may be discarded. Similarly, if the application requires a progressive transmission, only representation models allowing such a feature will be activated. The selection procedure may either be performed once in a supervised manner, and the profile pre-defined for each application, or it can be realized automatically according to a list of requirements attached to the application.

This first step of *a priori* selection defines a set of solutions in which the system is allowed to search for an appropriate coding scenario. Each coding strategy belonging to the set of admissible solutions is a very particular way of representing a set of data and results in different characteristics such as rate, distortion, error signal statistics, number of segmented regions, and so on. The second step is therefore to point out

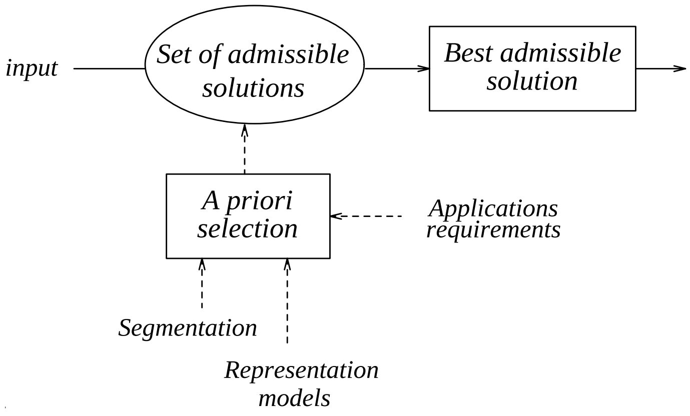

Figure 6: Dynamic coding procedure. The first step defines the set of admissible solutions with respect to the application, whereas the second step identifies the best admissible strategy.

the coding scenario which is optimal with respect to a predefined criterion. This requires the definition of a criterion together with a procedure for determining the optimal solution. However, this procedure is transparent to the decoder.

3 Dynamic Coding of Video - An Illustrative Example

Dynamic coding is a global approach to visual data representation and many variations on the same theme are possible. If we refer to the dynamic coding data representation system, any particular compression algorithm which segments the data and describes each data portion with a particular representation model can be viewed as a particular profile of dynamic coding. Specific profiles differentiate themselves by their particular set of admissible solutions and the criterion with respect to which the optimal admissible solution is defined. This section aims at providing an illustrative example of the dynamic coding approach, a video compression system for video-telephone/conference applications, based on the principles presented in the previous section.

As summarized in Fig. 6, the first operation consists of delimiting the set of admissible solutions with respect to the requirements imposed by video-telephone/conference applications. Section. 3.1 describes the set of admissible coding strategies and justifies the choices. Once the set of solutions is given, the best coding strategy must be identified. The procedure for determining the optimal solution is described in section 3.2. In this example, the optimal admissible coding strategy is defined as the one which minimizes the distortion subject to a maximum rate constraint. The procedure jointly optimizes the frame segmentation and the representation model associated to each segment [11]. As further explained in section 3.3, the optimization procedure leads to a multi-criterion frame segmentation which differentiates the system from classical object-oriented compression algorithms by the opportunity to define adaptively the notion of object. Emerging multimedia applications demands for object-oriented functionalities, enabling object manipulation at the bitstream level. Section 3.4 shows that the system is also capable, with minor modifications, of generating scalable bitstream made up of as many independent bitstream portions as there are specified objects.

3.1 Set of Admissible Solutions

The rest of the paper is devoted to illustrate the concept of dynamic coding in the framework of video-conference/telephone applications. As depicted in Fig. 6, the first step is to define the set of admissible solutions with respect to this type of applications. The full-duplex nature of communications involved in video-telephone/conference calls for low encoding/decoding delay. Moreover, very high compression is required owing to narrow channel bandwidth supporting these applications. Accordingly, the set of coding strategies is confined to solutions which fulfill two major requirements, namely, low coding delay and very high compression.

In order to achieve low coding delay, the segmentation is restricted to two-dimensional regions. This means that the video stream is viewed as a sequence of images rather than as a three-dimensional data set. Consequently, processing and compression of the video signal is performed on a frame by frame basis. The segmentation is further limited to quadtree partitioning. The motivation is twofold. First, quadtree segmentation constitutes a good trade-off between low complexity and representation cost. Second, the hierarchical structure of the quadtree segmentation greatly alleviates the succeeding problem of the optimal solution selection, as will be seen in the next section.

In the perspective of maintaining a reasonable encoding/decoding delay, representation models exploiting forward temporal redundancy are *a priori* discarded. Only spatial (intra) and backward temporal redundancy removal are permitted. Five representation models in agreement with these requirements and acknowledged as effective for high compression have been selected.

These representation models are briefly described hereafter as we assume that most readers are familiar with these techniques:

- *Motion compensation model*: consists in performing a half pixel accuracy backward motion estimation and compensation between the current region and the previously decoded frame. A translational motion model is assumed. Error resulting from the motion compensation is computed and encoded using the discrete cosine transform (DCT).

- *Background model*: is a degenerate case of the previous model where a region is compensated by the region at the same location in the previously decoded frame. No prediction error coding is performed.
- *Text and graphics model*: consists in representing a region as binary data. The original region is mapped into a segment displaying only two different grey-level values. Original grey-level values are discriminated into two clusters and set to the value of their corresponding centroids [12]. The resulting two-level region is represented by the two quantized centroid values and a binary mask locating the two clusters. This model is specially appropriate for video-conference where text or graphics may form a significant portion of the scene.
- *DCT model*: describes a region by its discrete cosine transform. The DCT coefficients are linearly quantized according to one of the available quantization tables. Only the non-zero quantized coefficients are transmitted while the location of zero coefficients is specified by run-length coding performed along a zig-zag scanning pattern.
- *Fractal model*: Each region is expressed as a contractive transformation of another part of the picture. Readers interested in a more detailed description of this model are referred to [13].

Within the fractal model, the graphic-oriented technique and the background model, the description of a given region is unique, since the quantization of the parameters is fixed. By contrast, the DCT and the motion compensation models may result in different possible region approximations since different quantization tables can be utilized. Throughout the rest of the paper, we will consider that there are as many representation models associated to the DCT and motion compensation modes as there are quantization tables. This terminology is slightly abusive but will greatly simplify the understanding of the approach.

These representation models constitute the upper limit of possible models associated to each region. However some of these models can be *a priori* discarded either on a frame basis or even at a region level. For instance, temporal modes must be discarded when compressing a reference frame (for which temporal prediction is prohibited). Similarly, some models may not be appropriate for small regions and can be *a priori* discarded at a region level.

The set of admissible solutions includes all possible quadtree frame segmentations together with one representation model associated to each resulting region. The cardinality of the set is very large and grows as $O(K^{4^d})$ as a function of the maximal allowed tree depth d. However, it is possible to build, at reasonable computational cost, the spine of the set of solutions thanks to its hierarchical structure. The spine is constructed as follows. Starting from the entire picture, the image is recursively split into four equally-sized subblocks. At each recursion step, the descriptions of the corresponding blocks are computed with respect to each of the representation models activated at this level of the tree. The recursion stops when the tree attains the maximally allowed tree depth[1]. This operation generates a complete tree where to each node is associated the parameters of the description within the five considered representation models. This spine allows a fast evaluation of the characteristics of any admissible solution. Indeed, the set of admissible solutions corresponds to all possible subtrees together with one representation model associated to each terminal node. Given the specification of a particular subtree, along with the specification of the representation model at each terminal node, we can rapidly evaluate the solution characteristics from the pre-computed spine. This step simply requires the combination of the specified representation models associated to each region of the solution.

Since the objective is to determine the optimal solution in a rate-distortion sense, these two values must be computed for each node with respect to each representation model in use. As required by the optimization procedure (see next section), the distortion must be evaluated not only over the entire block but also for its quadrants. The distortion can be readily evaluated by measuring the distance between the original data portion and its approximation with respect to a given model. The ideal distortion measure is obviously the subjective visual quality but no objective function has been acknowledged as encompassing the human visual system characteristics. The distortion measure used in this work is the popular square-error criterion given by:

$$D(I,\bar{I}) = \sum_{i=0}^{N}\sum_{j=0}^{M}(I(i,j) - \bar{I}(i,j))^2, \tag{1}$$

where I is the original data, $\bar{I}$ its decoded version and N, M are the vertical and horizontal size of the data segment.

The rate is computed as the sum of the cost of each parameter necessary for the description of the block.

[1] The maximal tree depth depends on parameters such as maximal allowed complexity, picture resolution and targeted bitrate.

In many cases, the cost attached to each parameter can be readily evaluated except if an *adaptive* entropy coder is employed. In this case, the current probability table associated to each parameter must be known along with the cost of each parameter estimated on the basis of its probability. A fair estimation of the resulting cost for a given parameter is given by its self-information:

$$R_p(a) = -log(P_p(a))/log(2), \tag{2}$$

where $R_p(a)$ is the estimated rate of parameter p whenever it takes the value a and $P_p(a)$ the corresponding probability.

The computational complexity of the entire system is mostly concentrated in the construction of the spine of the set of solutions. For each block in the tree, the algorithm must compute its description within the different available representation models. As the number of nodes at level k in the tree is 4^k, the total computational complexity associated to the construction of a tree of maximal depth d is given by:

$$\mathcal{C}(d) = \sum_{k=0}^{d} 4^k c(k), \tag{3}$$

where $c(k)$ is the complexity associated to one node at level k. and directly related to the computational complexity of the description of a block within each representation model as the sum of individual complexities:

$$c(k) = \sum_{i=1}^{n(k)} c_i(k) \tag{4}$$

where $n(k)$ is the number of available representation models at level k and $c_i(k)$ is the computational complexity of the description of a block at level k within representation model i. The complexity of the construction of the set of solutions is therefore adjustable by controlling the maximal tree depth d, by discarding some representation models at certain level of the tree to reduce $c(k)$, or by decreasing the individual complexities $c_i(\cdot)$ associated to each representation model. The possibility to discard some representation models from certain regions based on the statistics of the block is currently under investigation. For instance, the fractal model could be a priori discarded from regions identified as fine textures. This preprocessing step would result in a dramatic reduction of the computational complexity while maintaining the performance of the system.

3.2 Optimal Admissible Solution

The set of admissible solutions includes all the different strategies that can be used to represent the current frame. The remaining task now is to determine the optimal solution with respect to a predefined criterion. It requires to define a criterion with respect to which optimality is defined and a procedure for the determination of the optimal solution.

In the context of source coding, the traditional framework is given by Shannon's rate-distortion theory [14] where the overall source distortion is minimized subject to a channel rate constraint. Among admissible strategies, the goal is to determine the coding scenario minimizing the distortion provided that a given bitrate budget, R_{budget}, is not exceeded. Mathematically, the optimization criterion may be formulated as follows:

$$\min_{B \in \mathcal{S}} D(B) \ subject\ to\ R(B) \leq R_{budget}, \tag{5}$$

where $\mathcal{S}$ is the set of admissible solutions, $D(B)$ is the distortion, and $R(B)$ the bitrate resulting from encoding according to the scenario $B \in \mathcal{S}$.

The optimal bit allocation problem has been extensively addressed in source compression literature [15, 16, 17, 18]. Early approaches used continuous optimization achieved on the basis of diverse models both for the input signal and the quantizers characteristics [15, 16, 18]. Recently, the problem of discrete quantizer set and arbitrary input have been addressed in the context of vector quantization [17] as well as wavelet packet decomposition [19]. In these works, the rate-distortion optimization relies on a fundamental theorem of Lagrange multipliers developed in the framework of optimal resources allocation theory [20].

The rate-distortion optimization technique consists of converting the constrained minimization of (5) into an equivalent unconstrained problem by merging rate and distortion through the Lagrangian multiplier λ. The unconstrained problem comes down to selecting the coding strategy which gives the minimum of the Lagrangian cost function expressed as :

$$J(\lambda) = D(B) + \lambda R(B). \tag{6}$$

For a given multiplier λ, the minimization of the cost function $J(\lambda)$ results in a solution $B^*(\lambda)$, an associated rate $R^*(\lambda)$, and a distortion $D^*(\lambda)$. It has been demonstrated [20, 17] that, as we sweep λ over positive values, the pairs ($R^*(\lambda)$,$D^*(\lambda)$) trace out the optimal rate-distortion curve. In particular, if for a given λ_c,

$R^*(\lambda)$ happens to coincide with the targeted rate R_{budget}, then $B^*(\lambda)$ is the solution of the constrained problem (5). Readers further interested in this theorem and its implications are referred to [20, 17]. Mathematically, the optimization follows a two-step unconstrained problem:

- $$\textit{Find } \lambda_c \textit{ such that } R(B^*) \leq R_{budget} \textit{ with } B^* \textit{ minimizing } D(B) + \lambda_c R(B) \tag{7}$$
- $$\textit{Find } B^* \textit{ minimizing } J(\lambda_c) = D(B) + \lambda_c R(B) \tag{8}$$

The optimization problem is solved by iterative minimization of $J(\lambda)$ for λ values converging to the appropriate λ_c. It therefore requires the minimization of $J(\lambda)$ for arbitrary λ arguments. We first describe in details this minimization procedure. Based on this procedure, the technique to determine λ_c and the corresponding rate-distortion optimal solution will be presented.

• **Minimization of $J(\lambda)$:** Our objective is to determine the *generalized quadtree* frame partition together with the coding models associated to each resulting cell minimizing $J(\lambda)$. The denomination *generalized* stands for a partition in which each block can be represented by one of 16 possible geometric configurations, as depicted in Fig. 7. As can be seen, generalized quadtree segmentation differs from mere quadtree partition in which only two possible configurations coexist (one parent/no child, no parent/four children).

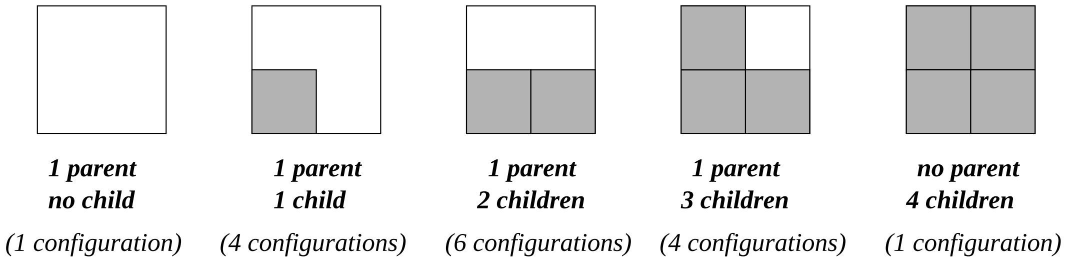

Figure 7: Possible configurations for representing a block. Dark areas denote block portions coded with the child model whereas white ones represent block portions coded with the parent model.

The minimization procedure is reminiscent to [19] where a similar set structure was involved though in a different context. The problem is greatly simplified thanks to the additivity property of J over distinct portions of the signal. Consider an image support partition $\{P_i, 0 \leq i < N\}$ of non-overlapping blocks P_i. Then, since $J(\lambda)$ is always positive:

$$\min(J(\lambda)) = \min(D + \lambda R) = \sum_{i=0}^{N-1} \min(D_i + \lambda R_i) = \sum_{i=0}^{N-1} \min J_i(\lambda), \tag{9}$$

where D_i and R_i are respectively the distortion and the rate associated to block P_i and $J_i(\lambda)$ the resulting cost function. The signification of Eq. 9 is that, given a partition $\{P_i, 0 \leq i < N\}$, the minimization of the cost function $J(\lambda)$ over the entire frame can therefore be performed independently over each segment P_i. Since this property is valid for any picture partition, optimal coding models can be determined by minimizing $J_i(\lambda)$ independently on each cell P_i.

However, the frame partition is itself a variable in the optimization procedure. The hierarchical structure of the set of solutions enables us to determine the optimal frame segmentation together with the optimal representation model associated to each segment. Each node in the tree symbolizes a block of a certain size and location. Subtrees attached to a given node correspond to different ways of representing the associated block. At the limit, the root node corresponds the entire image and all subtrees attached to the root correspond to the different ways of representing the whole picture.

Let us consider a particular block or node in the tree and assume the knowledge of the best option for representing its children. The block can be represented by the parent model as a whole or as the conjunction of the best options for each of the four children , or as any combination of parent model and children models as depicted in Fig. 7. For each of the sixteen configurations, we must determine the optimal parent coding model. The parent coding model optimal for the entire block may not be optimal over a sub-portion of the block. In other words, for grey-colored portions of block in Fig. 7, the optimal option is known but we still have to determine the best parent representation model for white-colored portion of the block. Therefore, given n active parent representation models, $15 \times n + 1$ options must be tested. For each option the Lagrangian function must be evaluated and compared to the current best option. The distortion is computed as the sum of the distortions for each independent quadrant when approximated by the representation model specified by the option. The rate is evaluated as the sum of the rates associated with each representation model activated, either as parent model or as child model. Moreover, in order to take into account the cost associated with the segmentation, the amount of bits necessary to describe the geometrical configuration (1 out of 16) must be included in the rate estimation of a given option.

Thanks to the hierarchical structure of the tree, the optimal coding strategy associated with a given frame can rapidly be determined by means of dynamic programming [21]. Optimization is performed recursively starting from bottom and by climbing up the tree to its root. The recursion starts with the parents of the leaves. The characteristics of the best coding option are assigned to the node as the best way to represent the corresponding region. The bottom-up approach allows, at each recursion step, to compare only $15 \times n + 1$ options. Rate-distortion characteristics of a configuration can be rapidly computed owing to the additive property of these quantities. The operation stops when the root of the tree is reached. At this point, the partition, together with its associated coding models which minimize $J(\lambda)$, are identified for a given quality factor λ. The distortion and rate corresponding to this solution are also determined.

- **Determination of optimal strategy**

As expressed by Eq. (8), the determination of the optimal strategy first requires to find the Lagrangian factor, λ_c, such that the solution which minimizes $J(\lambda_c)$ leads to a rate corresponding to the desired rate budget. The factor λ_c can rapidly be identified, thanks to the monotonic nature of optimal solution rate $R(B^*)$ versus λ. Indeed, the Lagrangian multiplier λ plays the role of a quality factor and acts as a trade-off between distortion for rate. When λ increases the distortion of optimal solution increases whereas its rate decreases. The appropriate λ leading to $R(B^*) = R_{budget}$ can therefore be retrieved by intelligently searching over λ values by fast convex search such as the bisection technique [22]. The algorithm therefore consists of iteratively determining the solution that minimizes $J(\lambda)$ for λ values converging to λ_c. Rates associated with these intermediate solutions help to determine subsequent estimates for λ. The procedure stops when $\lambda = \lambda_c$ is reached such that $R(B^*_{\lambda_c}) \simeq R_{budget}$.

The remaining task, as expressed by Eq. (8), is to minimize $J(\lambda_c)$ for the appropriate λ_c. This task is trivial since it consists of minimizing $J(\lambda)$ for the particular argument $\lambda = \lambda_c$. This last procedure leads to the frame partition together with optimal associated representation models, which give the minimum distortion without exceeding the rate budget R_{budget}.

Although in the given example, the distortion is minimized given a maximum rate budget, the algorithm is capable to find out the strategy that gives the minimum rate for a desired quality. We simply swap the rate and distortion functions in Eq. (5), which becomes,

$$\min_{B \in \mathcal{S}} R(B) \; \textit{subject to} \; D(B) \leq D_{target}, \tag{10}$$

where D_{target} is the limit of tolerable distortion. Equation 10 signifies that the rate is minimized subject to a maximal overall distortion. The dual Lagrangian cost function becomes $\bar{J}(\lambda) = R + \lambda D$. Since all the properties of $\bar{J}(\lambda)$ remain valid, the optimization procedure can be conducted without any modification by simply inverting the rate and distortion functions. The system is therefore capable of switching from a rate control to a distortion control mode.

The computational complexity of the entire optimization procedure is quite affordable. It must be understood that, given the spine of the set of solutions, the optimization procedure only manipulates pre-computed data and performs few additions and comparison tests. The determination of the appropriate Lagrangian multiplier λ requires in average ten iterations. Each iteration minimizes the Lagrangian cost function over the entire tree. A tree of depth d contains $(4^d - 1)$ nodes and for each node $15 \times n + 1$ configurations must be tested (where n is the average number of activated representation models). Computation of the Lagrangian function of one configuration requires only five additions and one multiplication since rate and distortion data are directly accessible. For typical values of tree depth d, and number of representation models n, this leads to 4×10^6 multiplications and 2×10^7 additions.

3.3 Interpretation as a Multi-Criterion Segmentation

Conventional object-oriented coding methods rely on the premise that the scene to be coded can be described as an union of objects. The sense of *object* may vary from one particular approach to another but it is always defined *a priori* and its definition conditions the entire compression procedure. For compressing visual data, classical object-based coding methods proceed in two distinct steps :

1. Scene analysis and object extraction. Typically, the data is segmented into uniform regions with respect to a criterion chosen as representative of the *a priori* definition of object.
2. Independent compression of every object. The compression technique remains the same for all the regions and is chosen to be particularly effective for the class of objects in question. This two step process is depicted in Fig. 8.

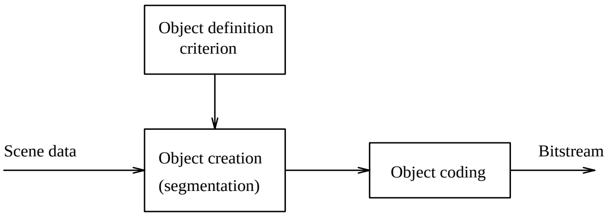

Figure 8: Schematic description of conventional object oriented coding methods

As an example, second-generation coding algorithms define an object to be any set of adjacent pixels having similar grey-level intensities [23]. Accordingly, the image is segmented into nearly uniform grey-level regions, and each region is ultimately approximated and encoded by polynomial fitting or some other technique appropriate for describing nearly uniform regions. As an additional example, some object-based image sequence compression techniques define an object as a set of adjacent pixels having similar motion [24]. In this case, the algorithm segments the frames into regions having coherent motion (based on the information provided by the motion vector field), and each region is approximated by motion compensation on the basis of one unique motion vector.

In contrast to classical object-oriented approach, dynamic coding offers the possibility of considering, *jointly*, the data segmentation and the region description. It is possible to design coding algorithms which inherently and adaptively define, *locally*, the notion of object instead of using an *a priori* object definition. This is equivalent to a *multi-criteria* segmentation where the scene is now segmented into objects possibly of different families as opposed to objects of the same family. The illustrative coding technique presented in the previous section is an example of such a coding algorithm. This algorithm defines the criterion by which the objects are created from a library of criteria put in competition, as shown in Fig. 9.

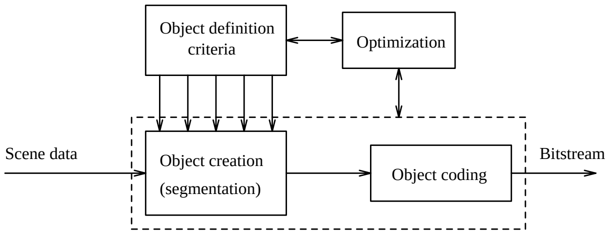

Figure 9: Schematic description of the proposed system

Since the definition of object is adaptive, the resulting regions created by this approach may not correspond to actual objects in the scene, from a computer vision point of view. This is illustrated in Fig. 10, where the different objects created by the algorithm are depicted together with their representation model for a particular frame. For instance, regions overlapping both a uniform background and part of the foreground are more efficiently represented as a whole by motion compensation rather than by segmenting the region further into background and foreground. Typically, a two-step object-based approach would have segmented the region into background and foreground and would have coded separately the two resulting segments. However, if object interactivity features are desired, it is possible to bring them into this scheme by special masking operations as demonstrated in the following section.

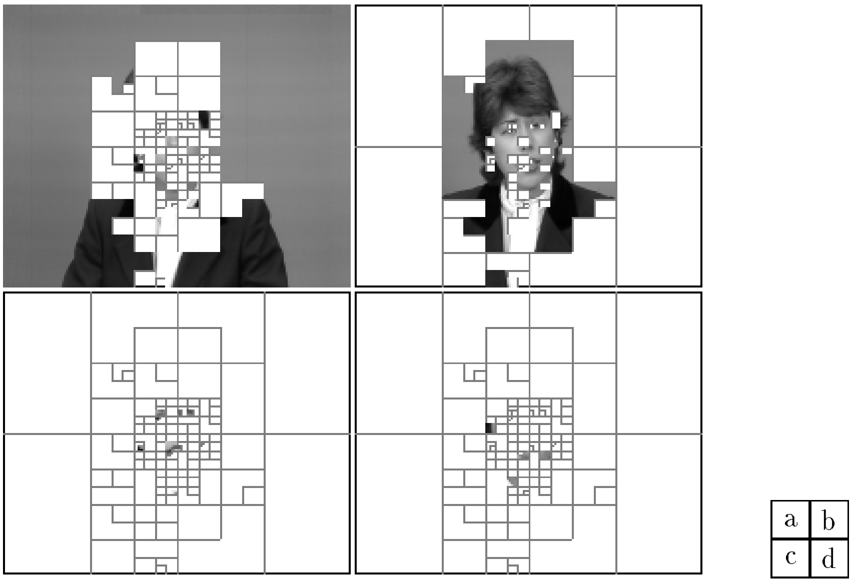

Figure 10: Example of frame segmentation (frame #86 Claire encoded at 8 kbit/sec.). Portion of the original image appears where a given mode is selected. (a) Background mode (b) Motion compensation mode (c) Fractal mode (d) Text mode

3.4 Object-Oriented Functionalities

Object scalability is the major functionality addressed by MPEG-4, and constitutes its specificity with regard to existing standards such as H.263. Designed for multimedia applications, MPEG-4 is expected to provide an object-oriented structured bitstream, thus allowing object manipulation, editing, browsing and modification prior to the decoding process as well as enabling content-based coding and transmission.

Despite the fact that the proposed algorithm results in a scene segmentation which generally differs from a semantic scene description, content-based interactivity functionalities can be easily incorporated. In this section, we demonstrate that the functionality of object-oriented bitstream scalability can be incorporated by minor extensions to the system. An object-based scalable bitstream is composed of as many independent portions as there are objects specified in the scene. The scene is therefore described object by object. This functionality enables us for example, to transmit and decode only specific objects in the scene, in order to obtain an absolute control over the amount of bits allocated to each object or to perform unequal error protection on the different streams depending on their relevance. The independent coding of a specific object exploits the generalized quadtree structure by exploring the solutions only in data segments that overlap the region of interest (ROI). In the recursive procedure of constructing of the set of solutions, two different options are now possible:

- If the block does not overlap the object, the recursion along this branch is stopped and the block is described using a specific representation model, namely, the *grey mode*. This representation model consists in replacing a block with uniform grey color segment. The description of a block using this model requires only one parameter (technique identifier). The resulting distortions associated with the block are set to 0.
- If a block overlaps the object, either partially or fully, only the pixels located inside the object are considered when computing the description of the block within the different representation models. Pixels outside the region are irrelevant and error in their approximation is not taken into account. Only distortions over pixels belonging to the object are computed.

The representation models must therefore be extended in order to allow the approximation of arbitrarily shaped regions and also in order to guarantee that portions of the scene not belonging to the object are not referenced, since they may not be available while decoding.

Extensions needed for object-based scalability affect only the procedure for constructing the set of admissible solutions. The only difference in the optimal solution selection phase is that the optimization now starts with an unbalanced pruned tree. Note that the distortion over the object is perfectly controllable since approximation errors over pixels located outside the object were set to 0. It is possible for instance to set a low limit of distortion to the most subjectively relevant object, and allocate the remaining available rate to the rest of the scene.

This approach allows us to obtain a perfectly object-scalable bitstream and enables the transmission and the decoding of the portion of the bitstream associated to the object of interest. The bitstream describing the scene is then made up of as many independent bitstream layers as there are specified objects. Fig. 11 illustrates the independent compression of two objects of the *Akiyo* scene, namely, the foreground and the background. Given a budget of 15 kbit to encode the entire frame in intra-frame mode, 8 kbit were allocated to the foreground object and 7 kbit to the background. Fig. 11 (a) gives the decoded picture when only the portion of the scalable bitstream corresponding to the foreground is decoded whereas Fig. 11 (b) gives the segmentation resulting from the encoding of the foreground object. As can be seen from Fig. 11 (a), the grey-level values of pixels lying outside the foreground object are arbitrary, since they are judged as irrelevant during the encoding process. This is confirmed by Fig. 11 (b) where we see that the recursive segmentation is stopped as soon as a block does not overlap at least partially the foreground area. Fig. 11 (c) and (d) provides the same results for the background object. The same comments apply.

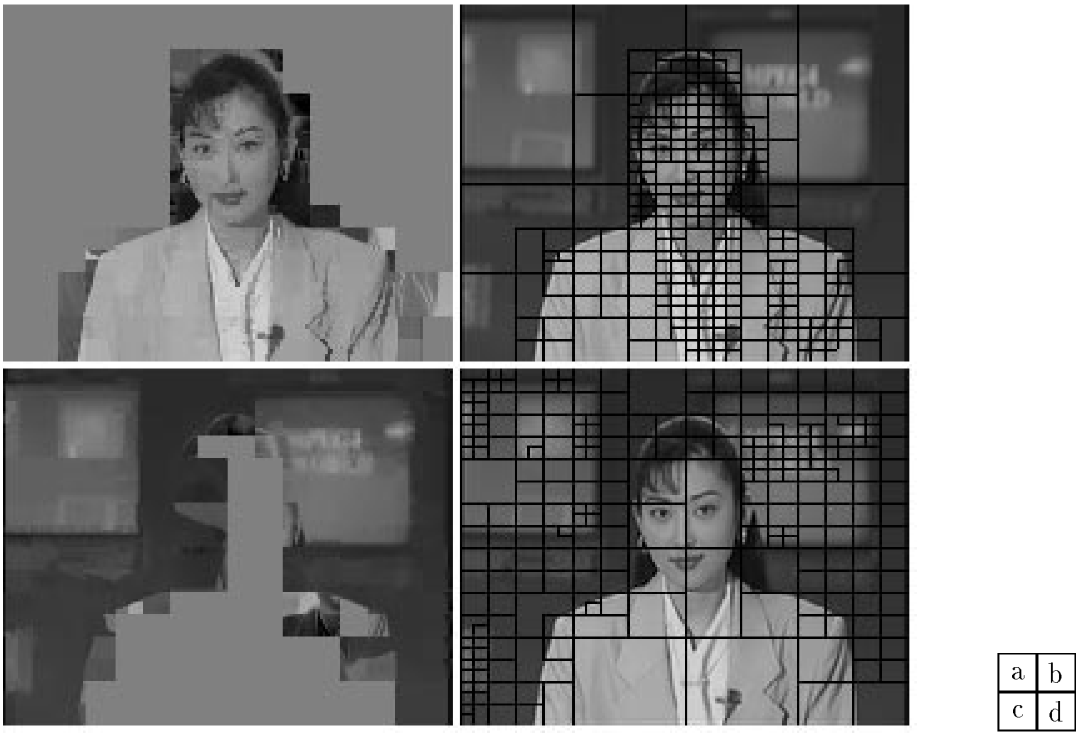

Figure 11: Object oriented coding (a) Foreground decoded picture (8 kbit) (b) Foreground resulting segmentation (c) Background decoded picture (7 kbit) (d) Background resulting segmentation

Fig. 11 shows that the shape and the location of the different objects comprising the scene cannot be retrieved from the bitstream. If this information is needed then the mask information locating the region of interest, must be transmitted to the decoder. The compression of the mask information is performed by a variant of the classical chain coding in which the redundancy between successive chain symbols is reduced by high order adaptive entropy coding [25]. If the mask information is made available, the blocks lying outside of the object of interest need not to be represented by the grey mode as the decoder can retrieve this information from the shape specification.

4 Simulation Results

4.1 Full Frame Compression Mode

Experiments have been carried out on two color video sequences, *Hall Monitor* and *News*. The input format is QCIF (144×176), 4:2:0, at 30 Hz. Input sequences have been temporally down-sampled to 5 Hz. Some frames extracted from the original sequences are shown in Fig 12 and 13. Simulations were performed on 10 seconds of the video signals.

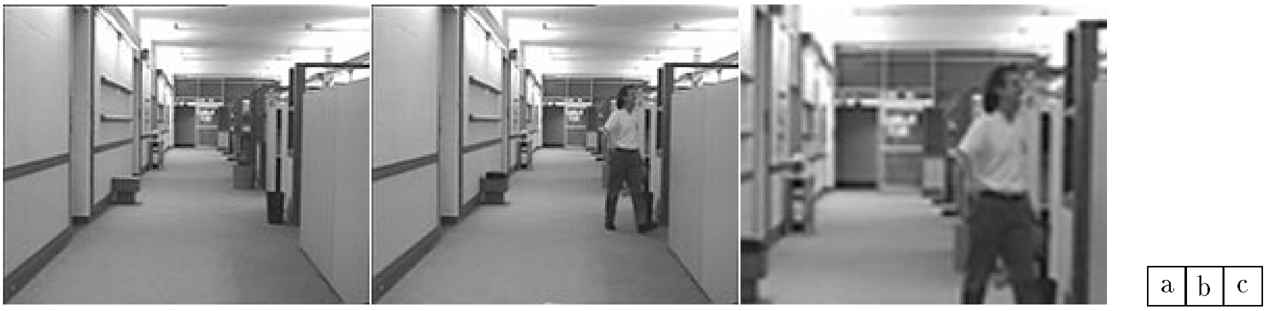

Figure 12: Original frames of *Hall monitor*. (a) Frame #1 (b) Frame #300 (c) Zoom on face area frame #300

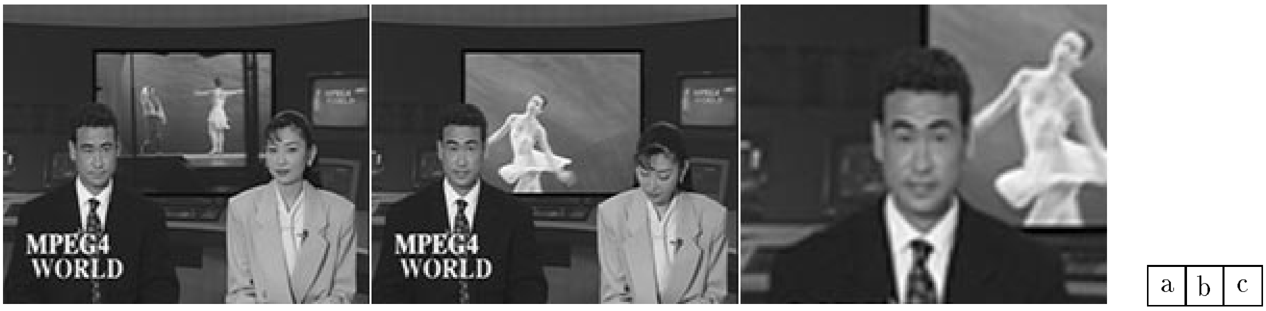

Figure 13: Original frames of *News*. (a) Frame #1 (b) Frame #300 (c) Zoom on face area frame #300

The system have been configured so as to reach a minimum distortion over each decoded frame in order to lead to a constant quality in the decoded sequences. In other terms, for each frame the rate is minimized subject to a given constant distortion. However, for each test sequence, the distortion values have been chosen in order to reach a specific overall bitrate. These distortion values leading to a specific bitrate have been determined experimentally. Bitrates of 10 and 48 kbit/sec. were targeted for the *Hall Monitor* scene whereas rates of 24 and 48 kbit/sec. were aimed at for the *News* sequence.

Experimental results are reported in Fig. 14 and 15. These plots describe the distribution of the bitstream around frames together with the peak signal-to-noise ratio (PSNR) behavior along the time axis. As expected, there are significant variations in the bitrate according to the local temporal and spatial scene complexity. Since experiments are performed at constant distortion, the PSNR behavior remains flat.

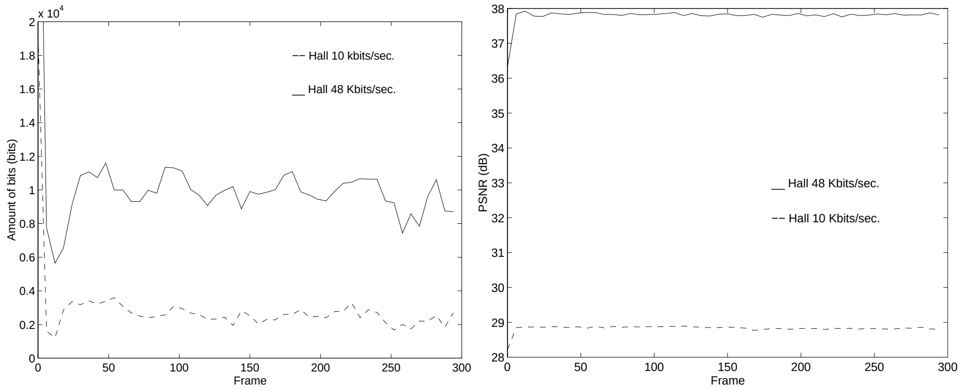

Figure 14: Rate distribution and distortion behavior for *Hall monitor*. (Left) Rate distribution at 10 and 48 kbit/sec. (Right) Distortion behavior at 10 and 48 kbit/sec.

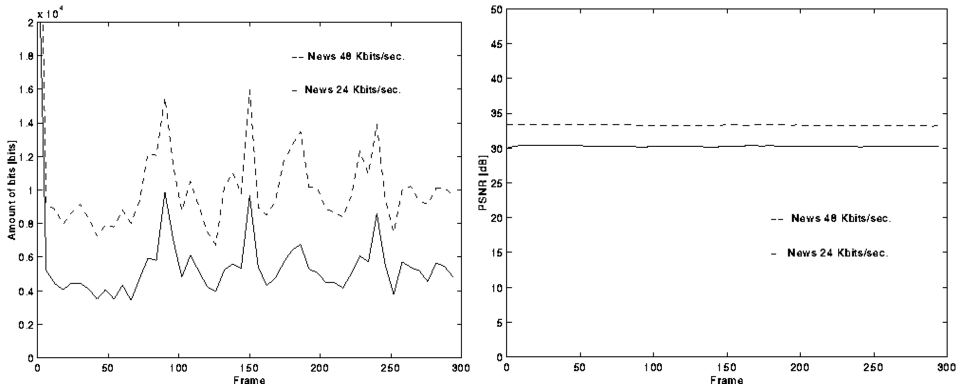

Figure 15: Rate distribution and distortion behavior for *News*. (Left) Rate distribution at 24 and 48 kbit/sec. (Right) Distortion behavior at 24 and 48 kbit/sec.

Fig. 16 to 19 gives some frames excerpted from the three reconstructed sequences for their two targeted bitrates. The most noticeable degradation is the loss of details in intricate regions (see zoom on face areas). However, edges remain sharp and motion rendition is natural. As at high compression, blocking artifacts may become visible, a post-processing algorithm in charge of reducing this type of degradation have been applied [26]. Interested readers are refered to [27] for more experimental results and performance comparisons with current state-of-the-art video compression systems.

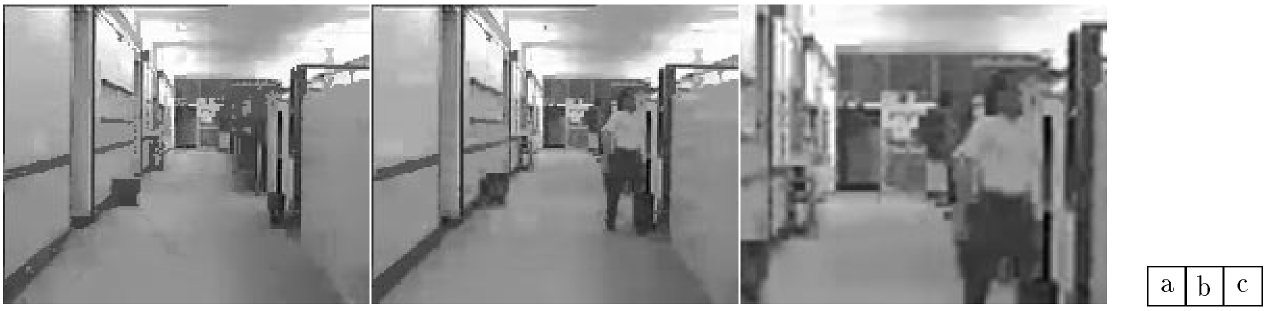

Figure 16: Reconstructed frames of *Hall monitor* at 10 kbit/sec. (a) Frame #1 (b) Frame #300 (c) Zoom on face area frame #300

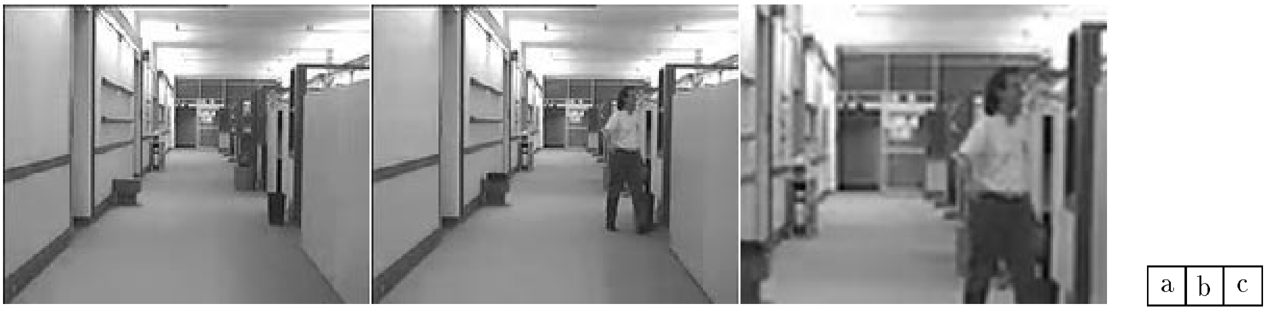

Figure 17: Reconstructed frames of *Hall monitor* at 48 kbit/sec. (a) Frame #1 (b) Frame #300 (c) Zoom on face area frame #300

4.2 Object-Oriented Scalability

To demonstrate the capability of object-oriented scalability, experiments have been performed on *Hall monitor* sequence. The same input format as the one of the previous collection of simulations has been used. Given a sequence of masks segmenting each frame into two objects, the system produces two independent bitstreams, each corresponding to a specific object of the scene. In these simulations, only two objects are considered namely the foreground (a man) and the background. The experiments reported here, again target a constant distortion on the foreground and a constant distortion on the background. These two distortions values were experimentally determined so as to lead to an overall bitrate of 48 kbit/sec. (namely 36.3 dB for the foreground and 36.8 dB for the background).

The results of this experiment are reported in Fig. 20. Fig. 20 (a) presents the bit distribution along the different frames and the different bitstream components. As can be seen, the object-oriented bitstream is made up of three layers: (1) the mask information, (2) the background compressed data and (3) the

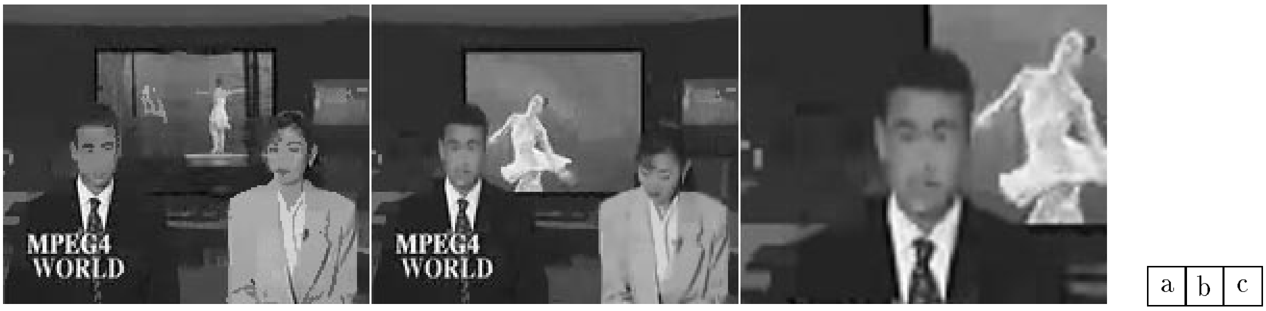

Figure 18: Reconstructed frames of *News* at 24 kbit/sec. (a) Frame #1 (b) Frame #300 (c) Zoom on face area frame #300

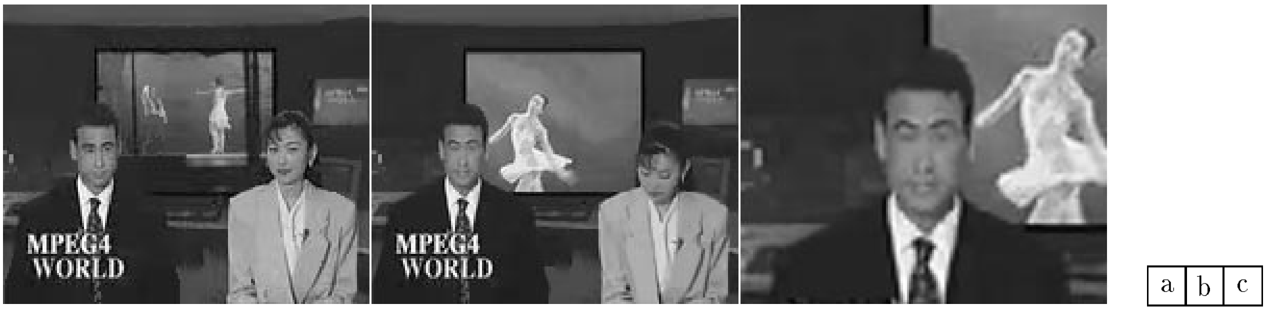

Figure 19: Reconstructed frames of *News* at 48 kbit/sec. (a) Frame #1 (b) Frame #300 (c) Zoom on face area frame #300

foreground compressed data. Note that the statistics related to the foreground object are only given starting from the frame #75 since this object first appears in the scene in this frame. Fig. 20 (b) depicts the distortion (PSNR) behavior of reconstructed frames along the time axis. The distortion values are computed over each object. The PSNR values resulting from the full-frame compression are also reported for the purpose of comparison. The variations in distortion originates from the discrete nature of the set of solutions. The small surface of the object further emphasizes this behavior.

As can be observed, the object-oriented configuration exhibits inferior rate-distortion performance when compared to the full frame compression configuration. However, since the object-oriented scheme enables a perfect control of quality and rate associated to each object, the increase of distortion have been distributed unequally among the two objects. The fidelity of the foreground reproduction remains similar whereas the background quality decreases by about 2 dB. Fig. 21 shows the decoded foreground and background objects of the frame #300 retrieved independently from the scalable bitstream. Fig. 22 gives some frames excerpted

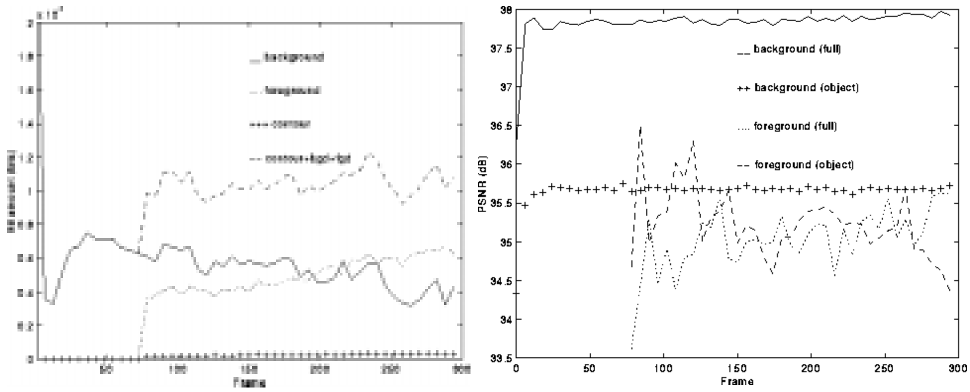

Figure 20: Rate distribution and distortion behavior for *Hall monitor* encoded at 48 kbit/sec. in the object-oriented configuration. (Left) Distribution of rate along the different bitstream components (Right) Distortion over the different objects composing the scene.

from the reconstructed sequence. The foreground and the background objects have been re-combined in order to reconstruct the entire scene.

The lower rate-distortion performance of the object-oriented configuration has multiple origins. First, in full frame compression mode, the quadtree segmentation is optimized in a rate-distortion sense. By enforcing an initial semantic segmentation, we move away from the frame partition optimal in terms of rate-distortion. Secondly, the description of the two objects are slightly cross-redundant. Eventually, even if the mask information constitutes a negligible portion of the bitstream, the amount of bits available to represent the image data decreases. The inferior performance in terms of rate-distortion behavior is the price to pay for object-oriented functionalities. The system is however capable of switching on the fly, from the object-oriented configuration to the full frame compression mode. This property permits one to employ the object-oriented mode only when strictly necessary.

Interested readers are referred to [28] for further experimental results both for full frame compression and for the object-oriented configuration.

5 Conclusion

This paper describes a novel approach to visual data compression called *dynamic coding.* Dynamic coding offers the opportunity to combine several compression techniques and segmentation strategies. Given a particular application, these two degrees of freedom can be constrained and assembled in order to produce a particular profile which answers the set of specifications dictated by the application. The basic principles

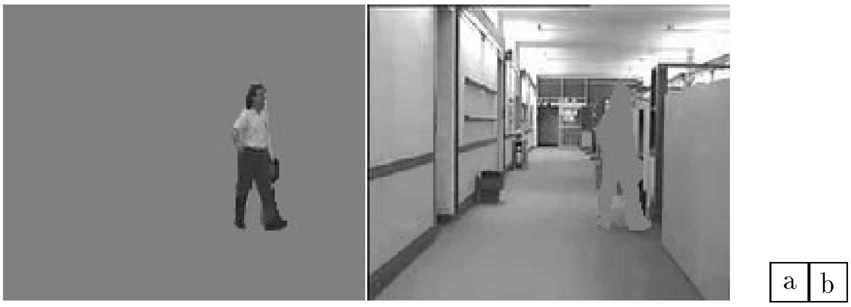

Figure 21: Reconstructed frames of *Hall monitor* compressed at 48 kbit/sec with the object-oriented configuration. (a) Frame #300 foreground object only, (b) Frame #300 background object only

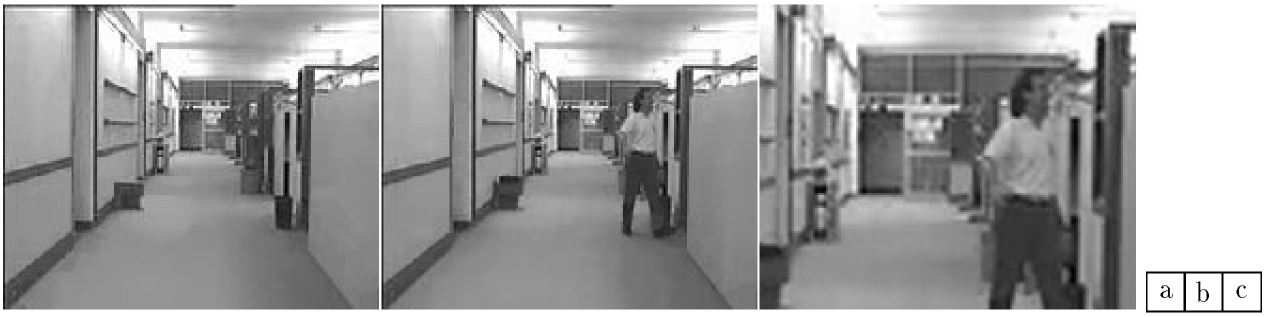

Figure 22: Reconstructed frames of *Hall monitor* compressed at 48 kbit/sec with the object-oriented configuration. (a) Frame #1 (b) Frame #300 (c) Zoom on face area frame #300

of this approach has been given together with the data representation system. Dynamic coding succeeds in combining *a priori* contradictory properties such as openness, flexibility, efficiency, and genericity in an optimal way.

In order to illustrate the concept of dynamic coding, an image sequence compression system, based on the principles described in Sec. 2, has been presented. Referring to Fig. 6, the set of admissible solutions has been constructed in the perspective of video-telephony and video-conference applications. Accordingly, the segmentation strategy has been restricted to a generalized quadtree partitioning and the collection of representation models has been confined to five compression techniques appropriate for very low bitrate applications. The best strategy is defined as the one which minimizes the distortion subject to a maximum rate budget or the inverse. The determination of the optimal solution relies on the theory of optimal resources allocation. Given the skeleton of the set of solutions, the system is capable of determining, at low computational cost, the optimal solution. The system operates a joint optimization of the segmentation and the representation model associated with each segment. This operation leads to a multi-criteria segmentation which enables an adaptive definition of objects. As the system performs a rate-distortion optimization, the control of the rate and of the quality of the decoded sequence is greatly simplified. In addition, the proposed scheme presents enough flexibility to provide, with very few modifications, object-oriented functionalities such as bitstream object scalability. Experiments have been carried out which demonstrate the power of the approach and the promises of the concept of dynamic coding.

References

[1] T. Ebrahimi and al. “Dynamic Coding of Visual Information”. Technical report, ISO/IEC JTC1/SC29/WG11/M0320, October 1995.

[2] SG XV CCITT. “Recommendation H.261 -video codec for audiovisual services at p*64kbit/s”. Technical Report COM XV-R37-E, August 1990.

[3] CCITT. “Description of ref. model 8 (RM8)”. Technical report, SG XV, June 1989.

[4] ISO/IEC JTC1/SC2/WG11 International Organization for Standardization. “MPEG Video Simulation Model Three”. Technical Report N 0010, July 1990.

[5] Motion Picture Expert Group. “Information technology - Coding of moving pictures and associated audio for digital storage media up to about 1.5 Mbit/s - Part 2: Coding of moving pictures information”. Technical report, CD 11172/ISO/IEC JTC1, 1991.

[6] “Motion Picture Expert Group, MPEG-II test model 4.2”. Technical report, ISO/IEC JTC1/SC29/WG11, 1993.

[7] ITU-T Expert group on very low bitrate video coding. “Simulation Test Model for Very Low Bitrate Image Coding (TMN3)”. Technical report, July 1994.

[8] Reusens, E. and Egger, O. and Ebrahimi, T. “Very low bitrate coding: which way ahead ? ”. In *IEEE Workshop on nonlinear signal and image processing*, pp. 1019–1022, Halkidiki, Greece, June 1995. invited paper.

[9] O. Egger, W. Li, and M. Kunt. “High Compression Image Coding Using Adaptive Morphological Subband Decomposition”. *Proceedings of the IEEE*, Vol. 83, No. 2, February 1995. invited paper.

[10] VM action group. "MPEG-4 Video Verification Model Version 1.1". Technical report, ISO/IEC JTC1/SC29/WG11/N1172, February 1996.

[11] E. Reusens. "Joint optimization of representation model and frame segmentatio n for generic video compression". *Signal Processing*, Vol. 46, No. 1, pp. 105–117, September 1995.

[12] L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data.* John Wiley and Sons, Inc., 1990.

[13] A. Jacquin. "Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformations". *IEEE Transactions on Image Processing*, Vol. 1, No. 1, pp. 18–30, January 1992.

[14] R.M. Gray. *Source Coding Theory.* Kluwer Academic Press, Norwell, MA, 1990.

[15] J.J. Huang and P.M. Schultheiss. "Block Quantization of Correlated Gaussian Random Variables". *IEEE Trans. Commun. Sys.*, Vol. CS-11, pp. 289–296, September 1963.

[16] A. Segall. "Bit Allocation and Encoding for Vector Sources". *IEEE transactions on Information Theory*, Vol. IT-22, No. 2, pp. 162–169, March 1976.

[17] Y. Shoham and A. Gersho. "Efficient Bit Allocation for an Arbitrary Set of Quantizers". *IEEE transactions on Acoustics, Speech, and Signal Processing*, Vol. 36, No. 9, pp. 1445–1453, September 1988.

[18] N.S. Jayant and P. Noll. *Digital Coding of Waveform.* Prentice-Hall, Englewood Cliffs, NJ, 1984.

[19] K. Ramchandran and M. Vetterli. "Best Wavelet Packet Bases using Rate-Distortion Criteria". *IEEE Int. Symposium on circuits and systems*, Vol. 2, pp. 971–974, May 1992.

[20] H. Everett. "Generalized Lagrange Multiplier Method for Solving Problems of Optimum Allocation of Resources". *Operation Research*, Vol. 11, pp. 399–417, 1963.

[21] A.J. Viterbi and J.K. Omura. *Principles of Digital Communication and Coding.* McGraw-Hill Book Company, 1979.

[22] W.H. Press, S.A Teukolsky, Vetterling W.T., and Flannery B.P. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, Cambridge, 1992. Second Edition.

[23] M. Kunt, A. Ikonomopoulos, and M. Kocher. "Second Generation Image Coding Techniques". *Proceedings of the IEEE*, Vol. 73, No. 4, pp. 549–575, April 1985.

[24] M. Hotter. "Object–Oriented Analysis–Synthesis Coding Based On Moving Two–Dimensional Objects". *Signal Processing: Image Communication*, Vol. 2, No. 4, pp. 409–428, December 1990.

[25] F. Bossen and T. Ebrahimi. "Region Shape Coding, ISO/IEC JTC1/SC29/WG11/M0318". Technical report, November 1995.

[26] R. Castagno and G. Rampani. "A Nonlinear Adaptive Technique for the Removal of Blocking Effect in Image Sequences Coded at Very Low Bitrate". Technical report, ISO/IEC JTC1/SC29/WG11/M0358, October 1995.

[27] H. Peterson. "Report of ad hoc group on mpeg-4 video testing logistics". Technical report, ISO/IEC JTC1/SC29/WG11/W1056, October 1995.

[28] T. Ebrahimi and al. "Dynamic Coding of Visual Information: Improved Implementation". Technical report, ISO/IEC JTC1/SC29/WG11/M0573, January 1996.