Submitted to the IEEE Trans. on Circuits and Systems for Video Technology

Optimal Data Partitioning of MPEG-2 Coded Video[1]

Alexandros Eleftheriadis and Dimitris Anastassiou

Department of Electrical Engineering
Columbia University, New York, NY 10027

{eleft,anastas}@ee.columbia.edu

January 22, 1996

Abstract

We analyze the problem of optimal data partitioning of MPEG-2 coded video in an operational rate-distortion context. The optimal algorithm is characterized and shown to have high complexity and delay. A causally optimal algorithm based on Lagrangian optimization is proposed, that optimally solves the problem for intra (I) pictures, while it provides an optimal solution for predicted/interpolated (P/B) pictures when the additional constraints of causal operation and/or low-delay are imposed. A memoryless version of the algorithm, theoretically optimal for intra-pictures only, is shown to perform almost identically but with significantly less computational complexity. Finally, a fast, suboptimal algorithm using purely rate-based optimization is also proposed, and is shown to perform quite close (within 1 dB) to the causally optimal one. Experimental results are provided using actual MPEG-2 coded video bitstreams.

1 Introduction

The traditional problem in video coding for the past several decades has been that of compression: describe the signal with as few bits as possible. The signal is treated as a single waveform, with compression employing techniques such as transform and/or prediction [8, 10] and resulting in a lossy representation. In several cases, however, it is beneficial to segment the original signal into multiple parts, and handle each one independently. Such an approach was originally applied to speech using the so-called sub-band coding approach [10], which partitions the signal into multiple

[1]Presented in part at the IEEE Int'l Conf. on Image Processsing, Austin, Texas, November 1994.

frequency bands. The primary motivation is that, since the human aural system perceives the various frequency bands in different ways, one could apply different compression techniques to each of the sub-bands.

A more general application of this principle is the so-called pyramidal, or hierarchical approach [15]. Here the signal is again decomposed into a number of different layers, but now each layer represents a successive refinement of the previous one. During compression, each layer is formed by compressing the difference between the original signal and its reconstructed version up to the particular layer. Consequently, an equivalent point of view is that each layer represents the compression error of its immediately lower layers. This representation is again typically (but not necessarily) lossy, hence leaving a residual compression error. In sub-band coding approaches, the individual layers are orthogonal to each other (or approximately so, depending on the filter bank used). For this reason, compression of the different layers can occur in parallel.

The benefits and applications of both approaches are numerous. A key feature is that they can facilitate interoperability. A classical example from the analog world is that of monochrome and color television receivers: the base layer here is the luminance signal (the monochrome component), while the added layers are the chrominance. The two layers are transmitted separately (modulated at different frequencies), and hence allow monochrome receivers to process color signals by "decoding" only the luminance part.

A more modern example from the digital domain is compatible High Definition Television coding: a layered approach allows the base component to be compatible with standard resolution television receivers, and hence a single signal can be used to service both systems.

Another important feature of layered compression is that it can be made robust to channel errors. In particular, one can associate a better transmission environment for the base layer, and less protected ones for the higher layers. In some transmission environments this association is directly supported. In packet-based networks for example, it is possible to mark higher layer packets so that, when congestion is created in intermediate routers or switches, they are dropped first. In systems that are not capable of prioritized transmission, one can essentially emulate the same effect by using different levels of forward error correction. By utilizing more efficient error-correcting codes for the lower layers, one can ensure—given some assumptions about the channel "noise"—that the base layer will arrive intact at the receiver. Examples of such channels are over-the-air broadcast, as well as individual virtual circuits within packet-based networks.

A significant drawback of layered approaches is that they are, in general, less efficient than their single-layer counterparts. In other words, for a particular signal to noise ratio (SNR), it is better to compress a signal using a single layer than multiple ones. "Better" here implies that fewer bits are required to represent the signal. In several applications, however, this is acceptable due to the added flexibility. We should also note that layered approaches also tend to be much more expensive to implement that single-layer ones, due to the added encoding and decoding complexity.

From an applications perspective, the most important drawback of layering is that it is embedded in the encoding method. In other words, the layers can only be constructed during the encoding process; doing so at a later stage would require significant computational resources, in essence consisting of full decoding and recoding. There are several reasons, however, to desire a layered structure even without support from the encoder. First, due to the cost of hierarchical (or scalable) encoding, it is likely that single-layer, general-purpose encoders will dominate in actual systems. Second, the exact partitioning point in terms of bit rates is not obvious, due to the potentially large number of channel types over which the signal may be transported. Finally, due to the loss in compression efficiency it is likely that, in applications such as video-on-demand, only a single-layer high-quality version of the signal will be utilized (stored for retrieval by the users). It is then important to examine approaches in which layering is provided after encoding has already taken place.

In this paper we examine one such approach, called "data partitioning", which is applicable to any block-based, transform coding scheme. Our primary focus will be the MPEG-2 coding scheme [2], which provides direct support in its bitstream syntax to effect data partitioning, and in which partitioning was first introduced.

We analyze the problem of optimal data partitioning using an operational rate-distortion approach. The optimal algorithm is characterized, and is shown to have significantly high complexity and delay, as a result of the temporal structure of predictive compression. A “causally optimal” algorithm based on Lagrangian multipliers is described; it optimally solves the problem when the additional constraints of causal operation and/or low-delay are imposed. A memoryless version of the algorithm, theoretically optimal for non-predictive compression only, is shown to perform almost identically but with significantly lower computational complexity. Finally, a fast, suboptimal algorithm using “rate-based” optimization is also proposed, and is shown to perform quite close (within 1 dB) to the causally optimal one. We should note that data partitioning can be applied to potentially any compression scheme; although the theoretical tools would be identical to those presented here, the performance characteristics may be significantly different.

The structure of the paper is as follows. In Section 2 we introduce the problem of data partitioning, and formulate it in an operational rate-distortion context. In Section 3 we present the optimal solution for non-predictive coding, whereas in Section 4 we analyze the more general predictive coding case. In both cases we present experimental results using actual MPEG-2 video bitstreams. The paper concludes with a summary of the key results presented and a discussion of their importance for actual applications. It is assumed that the reader is familiar with the algorithmic foundation and bitstream syntax of MPEG. An overview is presented in [13], while detailed descriptions can be found in [9, 14]; the standards themselves are documented in [1] (MPEG-1) and [2] (MPEG-2).

2 The Data Partitioning Problem

2.1 Data Partitioning

Data partitioning is a feature of the MPEG-2 standard that provides for the segmentation of a coded signal bitstream into two components or partitions [2, 5, 6, 9]. It can be a very effective tool for the transmission of video over channels that allow selective protection of each of the partitions. Channels of this type can be implemented, for example, using increased forward error correction, or employing high priority transmission in an ATM-based networking environment. By transmitting the most critical information with high reliability, i.e., over the highest quality channel, the average quality of the signal reconstructed at the receiver can be significantly increased for the same level of channel distortion. This feature is one of the major benefits of pyramidal or—more generally—hierarchical, multi-layer coding schemes.

An important characteristic of data partitioning is that it can be employed even after encoding has taken place, in contrast with other hierarchical approaches, such as the SNR, spatial, or temporal scalability modes of MPEG-2 [2, 9], or the embedded DCT coding approach proposed in [17]. This is because the encoder does not need to maintain a prediction loop per each signal layer, a necessary requirement for a pyramidal scheme in which each coding layer is an enhancement of its previous one. As a direct consequence, it is also less robust in the sense that neither partition is self-contained; loss of information in either one will cause error propagation and accumulation during the decoding process if temporal predictive/interpolative modes are used. As we will see later on, error accumulation can in fact be kept under control. Although data partitioning is currently supported only in MPEG-2, it is trivial to incorporate into other coding schemes.

The system diagram of the data partitioning scheme is shown in Figure 1. In between an MPEG-2 encoder/decoder pair, the bitstream (assumed here to be coded at the constant rate of B Mbps) is split into two parts, each being transmitted on a different "channel". In this paper we assume that channel 0 is a perfect one, i.e., it exhibits no losses, errors, or insertions. We

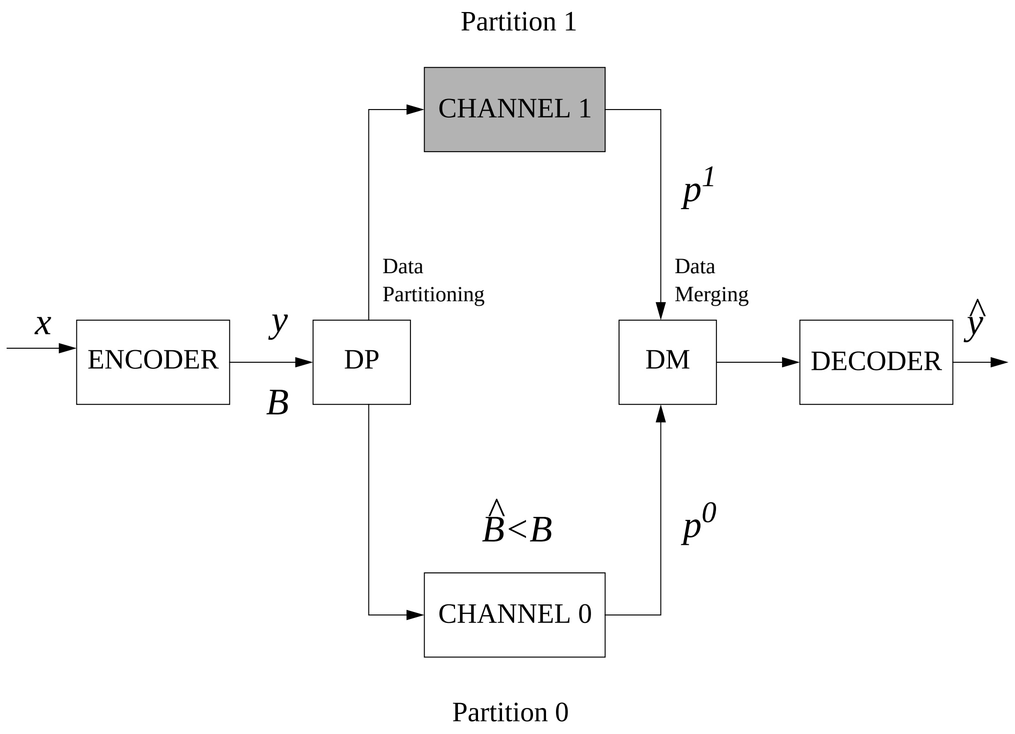


Figure 1: Block diagram of a Data Partitioning system.

also assume that it has a given fixed available bandwidth $\hat{B} < B$. Channel 1 is assumed to exhibit arbitrary stochastic behavior (a minimax problem formulation will make the details of this behavior irrelevant).

In other words, we are given a bitstream of bit rate B, but our communication resources only allow us to reliably transmit at a bit rate of $\hat{B}$. The problem is then how to optimally split the bitstream into two parts, the base one complying with the rate constraint $\hat{B}$, so that the quality of the decoded signal at the receiver is maximized.

Partitioning is performed at well-defined points in the bitstream syntax [2], called *breakpoints*. These can occur at various levels of the bitstream hierarchy. For our purposes, and to ensure that partition 0 is independently decodable, we will constrain the allowable breakpoint positions so that critical quantities such as macroblock address increment (indicating the relative position of a macroblock with respect to the previously coded one) and DCT DC differential values (for intra-coded macroblocks) are included in partition 0. As a result, partitioning will only affect the

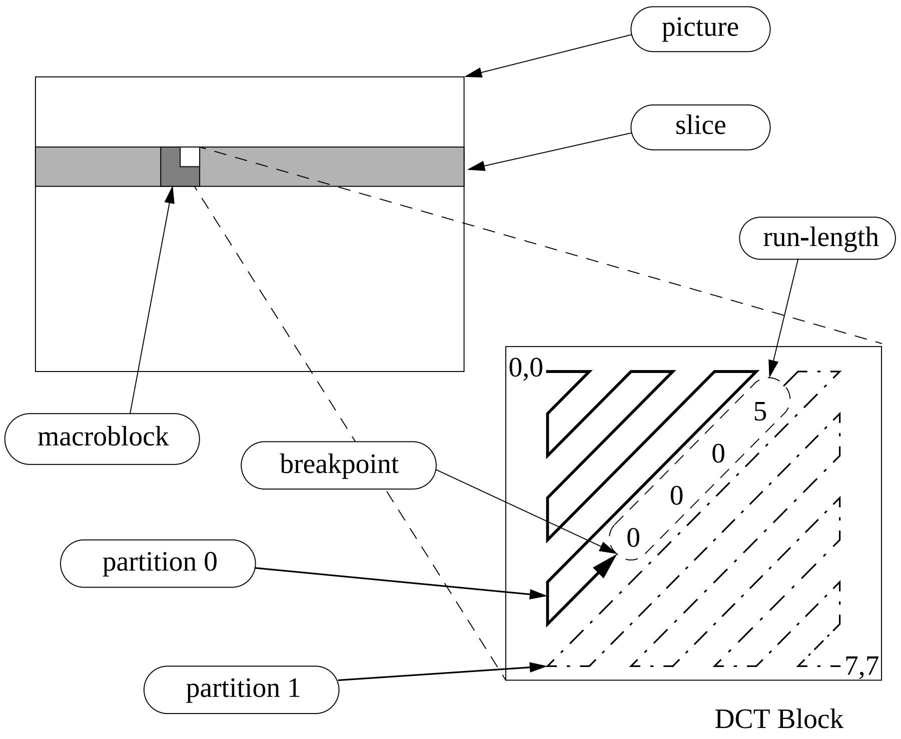


Figure 2: Breakpoint position in the zig-zag pattern of DCT coefficients.

number of coefficient run-length codes that will be carried in partition 0, while the rest will be assigned to partition 1. This is depicted in Figure 2.

Note that, in MPEG-2, the breakpoint value is the same for all blocks of a given slice. The breakpoint value, i.e., a fixed-length code indicating the number of run-length codes that are included in partition 0, is included in the slice header. Sequence headers are replicated in partition 1 to increase robustness, and hence the total rate for the transmission of the signal is slightly increased. In partition 1, the breakpoint value is set to 0 since it is not needed. We now proceed to a mathematical formulation of the problem.

2.2 General Problem Formulation

Denoting by y the coded video signal, by $\hat{y}$ the output of the decoder, by p^i the signal of the i-th partition, and by $R(\cdot)$ the bit rate, the problem of optimal data partitioning can be expressed as follows:

$$\min_{R(p^0) \leq \hat{B}} \{\|y - \hat{y}\|\} \tag{1}$$

The metric $\|\cdot\|$ above denotes the squared error criterion:

$$\|\mathbf{x}\| \equiv \mathbf{x}^T \mathbf{x} = \sum_{i=0}^{N-1} x(i)^2 \tag{2}$$

and is applied only in the luminance component. The rate constraint, however, refers to all three color components. Since channel 1 is assumed to exhibit stochastic behavior, we consider the deterministic problem of minimizing the *maximum average distortion* D, i.e.,

$$\min_{R(p^0) \leq \hat{B}} \left\{ D \overset{\text{def}}{=} \max \{\|y - \hat{y}\|\} \right\} \tag{3}$$

This corresponds to the case where the entire partition 1 is lost. D will be referred to in the sequel as the "partitioning distortion".

The optimization window in (3) is not specified, and it can span from just a part of a picture, up to any number of pictures. In general, and taking into account that data partitioning as described here is performed after encoding has taken place, it is desirable to keep the end-to-end delay low. Computational complexity considerations impose additional constrains on the window length, as will be made evident later on. Consequently, we will typically be interested in solutions of (3) that consider up to a single complete picture.

An important aspect of the problem not readily evident in (3) is its recursive nature, caused by the corresponding recursive process with which y and $\hat{y}$ are generated (decoded) when P and B pictures are involved. In the following we separately consider the two cases: optimal data partitioning in non-predictive coding (I pictures only), and optimal data partitioning in predictive coding (I, P, and B pictures).

3 Non-Predictive Coding

3.1 Problem Formulation

In non-predictive or intra-picture only partitioning, there is no temporal dependence between pictures. Consequently, the partitioning distortion will simply consist of the DCT coefficients that were assigned to partition 1. Since the DCT matrix C is unitary, i.e.,

$$C^T C = I \tag{4}$$

the energy of the signal in the spatial domain is equal to its energy in the transform domain. In other words, and considering the one-dimensional case for simplicity, if:

$$\mathbf{X} = C\mathbf{x} \tag{5}$$

then

$$\mathbf{x}^T\mathbf{x} \equiv \sum_{i=0}^{N-1} x(i)^2 = \sum_{i=0}^{N-1} X(i)^2 \equiv \mathbf{X}^T\mathbf{X} \tag{6}$$

Now let b denote the truncation point, i.e., all DCT coefficients from b up to $N-1$ are moved to partition 1. Considering the truncated—in the DCT domain—representation $\tilde{\mathbf{x}}$ of $\mathbf{x}$ and using (6) we have:

$$\|\mathbf{x} - \tilde{\mathbf{x}}\| = \|\mathbf{X} - \tilde{\mathbf{X}}\| = \sum_{i=b}^{N-1} X(i)^2 \tag{7}$$

eq. (7) provides an expression for the truncation, or partitioning distortion directly in the DCT domain. Generalization to two dimensions is straightforward.

Let us now consider the partitioning distortion in two dimensions for a group of blocks. We recall that the breakpoint values are identical for all blocks of a given slice i (Section 2.1). This value will be denoted by b_i, and indicates that the b_i-th and higher-order DCT run-length codes of all blocks of this slice will be removed and placed in partition 1. The domain of b_i is the set of values $\{0, 1, \ldots, 64\}$. Since blocks are transformed independently, the partitioning distortion for a set of blocks will simply be the sum of the partitioning distortions of the individual blocks.

Denoting the DCT coefficient of the k-th run-length of block j of slice i by $X_j^i(k)$, and the set of blocks that belong to slice i by $\mathbf{S}_i$, we can express the partitioning distortion for a particular slice as:

$$D_i(b_i) = \sum_{j \in \mathbf{S}_i} \sum_{k \geq b_i} X_j^i(k)^2 \tag{8}$$

Note that this is a function of the breakpoint value b_i. Since error calculations are only done on the luminance component, we will assume that, for chrominance blocks,

$$D_i(b_i) = 0 \tag{9}$$

Returning to eq. (3), we can now explicitly express the problem of minimizing the maximum partitioning distortion D as:

$$\min_{R(p^0) \leq \hat{B}} \{\max \{\|y - \hat{y}\|\}\} \Longleftrightarrow \min_{\sum_{i=1}^{S} R_i(b_i) \leq \hat{B}} \left\{ \sum_{i=1}^{S} D_i(b_i) \right\} \tag{10}$$

where $R_i(b_i)$ denotes the rate required to encode slice i when the breakpoint value b_i is used, and S is the total number of slices considered (may span several pictures).

Our objective here is to find those values b_i^*, $i = 1, \ldots, S$ that minimize the maximum distortion as given in (10). An exhaustive search would be clearly impossible, as the number of possible combinations that would have to be examined can be huge (65^S). We recall that data partitioning is typically applied immediately prior to transmission (when the value of $\hat{B}$ becomes known), and hence complexity and delay considerations are very important.

3.2 The Optimal Algorithm

This constrained minimization problem can be solved using the approach of Lagrange multipliers [7]. A similar algorithmic approach but in a different context has been used in [11, 12, 18]. The Lagrange multipliers approach converts the constrained optimization problem to an unconstrained one, by adding more dimensions to the parameter space. Consider the following problem. Given a constraint B, find

$$\min_{b \in A} D(b) \tag{11}$$

subject to

$$R(b) \leq B \tag{12}$$

Then the following theorem holds [7].

Theorem 1 *For any $\lambda \geq 0$, the solution $b^*(\lambda)$ to the unconstrained problem*

$$\min_{b \in A} \{D(b) + \lambda R(b)\} \tag{13}$$

is also the solution to the constrained problem (11)–(12) with the constraint $B = R(b^*(\lambda))$, that is, with $R(b) \leq R(b^*(\lambda))$.

The proof is quite simple and can be found in [7]. Note that Theorem 1 does not guarantee any solution to the constrained problem (11)–(12) (in other words, the two problems are not equivalent). It only indicates that for every nonnegative λ, there is a corresponding constrained problem which solution is identical to that of the unconstrained one. If, however, $R(b^*(\lambda))$ happens to be equal to B, then $b^*(\lambda)$ is the desired solution for the constrained problem.

Since the constraint B in our problem is given (the reliable channel bandwidth $\hat{B}$), our algorithm will have to find an appropriate value for λ so that $R(b^*(\lambda) = B$. Since the domain of b in our case is discrete, such an exact solution may not be attainable. We will consequently be satisfied with a solution for which $R(b^*(\lambda))$ is as close as possible to B.

Returning to our original problem, we can rewrite (10) as an unconstrained problem as follows

$$\min \left\{ \sum_{i=1}^{S} D_i(b_i) + \lambda \sum_{i=1}^{S} R_i(b_i) \right\} \tag{14}$$

By defining the per-slice quantity

$$L_i(\lambda, b_i) \overset{\text{def}}{=} D_i(b_i) + \lambda R_i(b_i) \tag{15}$$

the above can be rewritten as

$$\min \left\{ \sum_{i=1}^{S} L_i(\lambda, b_i) \right\} \tag{16}$$

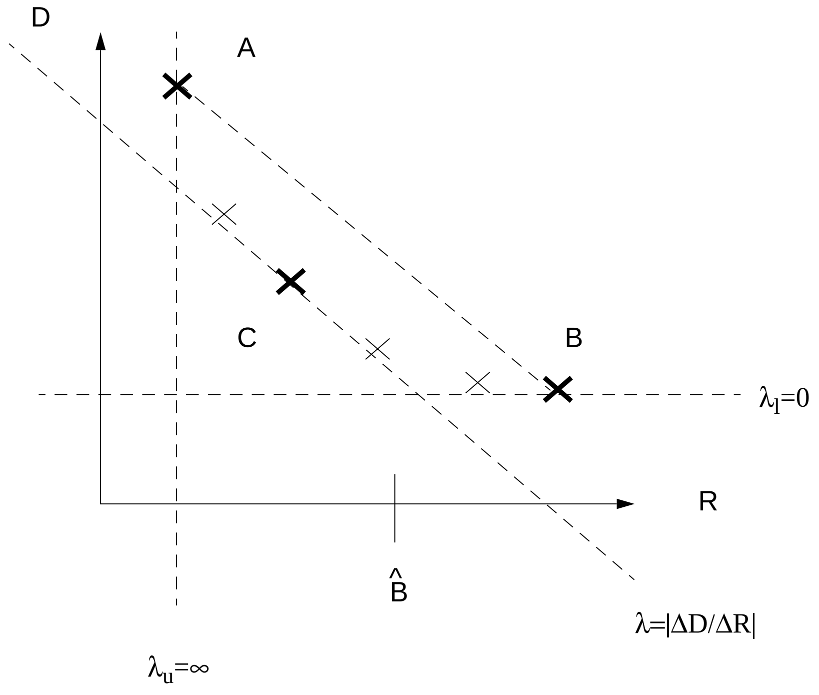


Figure 3: Overview of the bisection algorithm.

We observe that, given a particular λ, the minimization problem above can be solved independently for each slice, since $L_i(\lambda, b_i) \geq 0$. In other words, each L_i can be minimized independently of the others. This structure helps to significantly reduce the complexity of the problem. Within each particular slice, one can even use exhaustive search to obtain the optimum breakpoint value b_i^*, since the possibilities are limited (65 in the worst case). Hence the complexity of the problem becomes proportional to $65S$, where S is the number of slices (recall that for a full-search it is 65^S).

In order to find the optimal solution b_i^*, however, we must also find the appropriate value λ^* for λ. This can be accomplished using an iterative bisection algorithm [11, 12]. The algorithm starts with two initial estimates for λ (typically its extreme values 0 and ∞), and continuously refines its estimate until convergence is achieved.

Figure 3 illustrates the procedure for the simple case of a single slice. The graph shows the various rate-distortion points (marked with "x") when different breakpoint values are selected. For example, point A (first from the left) corresponds to the breakpoint value $b = 0$, i.e., to the case where all DCT coefficients are carried in partition 1. This gives rise to a particular partitioning distortion D and rate R that is required to represent the slice, as depicted in the figure. The rate is non-zero, since there are also overhead bit for headers etc. Similarly, point C corresponds to the breakpoint value $b = 2$. Since more DCT coefficients are included in partition 0, the rate is slightly increased but the distortion is reduced. Hence the $R(D)$ curve[2] is monotonically non-increasing. The curve does not necessarily have to have 64 points, since typically only a small number of run-lengths are needed to encode each block.

We should note that these $R(D)$ curves are not results of a stochastic minimization problem as in rate-distortion theory [3, 4], but discrete point curves that result from actual compression and differ from slice to slice. This is the reason why the term "operational" rate distortion minimization is used to differentiate it from the stochastic case (an $R(D)$ curve obtained from actual data is shown in Figure 4).

As initial values for λ we select the two extreme cases $\lambda_l = 0$ and $\lambda_u = \infty$ (the subscripts are for "lower" and "upper' respectively). In the former case the minimum is achieved by independently minimizing the distortion, and hence the optimal breakpoint for this value of λ (denoted by $b^*(\lambda)$) is obtained by using the maximum possible value of b: $b^*(0) = b_{\max} \leq 64$. This solution is indicated at point B in Figure 3. In the case $\lambda_u = \infty$, the minimum is achieved by independently minimizing the rate, corresponding to $b^*(\infty) = 0$ or point A of Figure 3[3]. At points A and B we also show the lines that pass from these points and have as a slope the negative value of their corresponding λ. Observe that these points minimize the expression: $D(b) + \lambda R(b) + c$, and hence for some value of the constant c (for the particular λ) the optimum solution is *on* the line, while all other points are above it.

[2]Although the term "$R(D)$ curve" is used in this paper, we should note that, strictly speaking, it is imprecise as it implies continuity.

[3]Using this formulation, and for purposes of precision, the rate should be exactly 0; one can always, however, subtract the overhead bit rate from the rate constraint $\hat{B}$, ensuring this way that $R(0) = 0$.

We observe that our initial estimates $R(\lambda_l)$ and $R(\lambda_u)$ contain the desired target rate $\hat{B}$, which ensures that the problem is feasible. The next step is to select a new value for λ, which can be done in any number of different ways. Lacking any a priori information on the $R(D)$ curve, and given its high variability from slice to slice, a plausible selection is the slope of the line segment interconnecting our original points A and B. We thus have:

$$\lambda_{\text{next}} = \frac{D(\lambda_u) - D(\lambda_l)}{R(\lambda_l) - R(\lambda_u)} \tag{17}$$

Next, we minimize $D(b) + \lambda R(b)$ for this particular λ, and obtain as a solution, say, point C. Note that the new optimal breakpoint value will be between those of points A and B.

We then examine which of the intervals

$$[R(b^*(\lambda_u)), R(b^*(\lambda_{\text{next}}))) \quad \text{and} \quad (R(b^*(\lambda_{\text{next}})), R(b^*(\lambda_l))] \tag{18}$$

contains our target bit rate $\hat{B}$, and repeat the procedure from the start using these new values of λ as starting points. If it turns out that $R(b^*(\lambda_{\text{next}})) = \hat{B}$ then we have found an exact solution. In practice, convergence will occur when the new $R(D)$ point coincides with one of the initial two points.

We now present the detailed description of the algorithm, applicable to any number of slices. Since in an actual implementation it is more convenient to deal directly with the number of bits instead of the rate, in the following we can consider that rate-related quantities refer to just quantities of bits. The two are proportional to each other, related by a normalization constant, and hence the two interpretations are equivalent.

We denote by $b_i^*(\lambda)$ the optimal breakpoint for slice i for the particular value of λ, and $R_i^*(\lambda)$ and $D_i^*(\lambda)$ the optimal rate and distortion respectively of slice i for the given value of λ, i.e., we have:

$$R_i^*(\lambda) \overset{\text{def}}{=} R_i(b_i^*(\lambda)) \quad \text{and} \quad D_i^*(\lambda) \overset{\text{def}}{=} D_i(b_i^*(\lambda)) \tag{19}$$

We also denote by R_{budget} the target bit budget for the particular set of slices $\{\mathbf{S}_i\}$. We should note that although the average rate of partition 0 has to be less than or equal to $\hat{B}$, there is flexibility on how the target bit budget for any given set of slices is allocated.

Lagrangian Minimization Algorithm

Step 1: Initialization

Set $\lambda_l = 0$ and $\lambda_u = \infty$. If the inequality:

$$\sum_{i=1}^{N} R_i^*(\lambda_u) \leq R_{\text{budget}} \leq \sum_{i=1}^{N} R_i^*(\lambda_l) \tag{20}$$

holds as an equality for either side, an exact solution has been found. If the above does not hold at all, then the problem is infeasible (this can happen if the target rate $\hat{B}$ is too small). Otherwise go to Step 2.

Step 2: Bisection and Pruning

Compute:

$$\lambda_{\text{next}} := \left| \frac{\sum_{i=1}^{N} [D_i^*(\lambda_u) - D_i^*(\lambda_l)]}{\sum_{i=1}^{N} [R_i^*(\lambda_u) - R_i^*(\lambda_l)]} \right| \tag{21}$$

and find $R_i^*(\lambda_{\text{next}})$ and $D_i^*(\lambda_{\text{next}})$ such that $B_i^*(\lambda_u) \leq B_i^*(\lambda_{\text{next}}) \leq B_i^*(\lambda_l)$.

Step 3: Convergence Test

If

$$\sum_{i=1}^{N} R_i^*(\lambda_{\text{next}}) = \sum_{i=1}^{N} R_i^*(\lambda_u) \quad \text{or} \quad \sum_{i=1}^{N} R_i^*(\lambda_{\text{next}}) = \sum_{i=1}^{N} R_i^*(\lambda_l) \tag{22}$$

then stop; the solution is $B_i^*(\lambda_u)$, $i = 1, \ldots, N$. If

$$\sum_{i=1}^{N} R_i^*(\lambda_{\text{next}}) > R_{\text{budget}} \tag{23}$$

then $\lambda_l := \lambda_{\text{next}}$, else $\lambda_u := \lambda_{\text{next}}$.

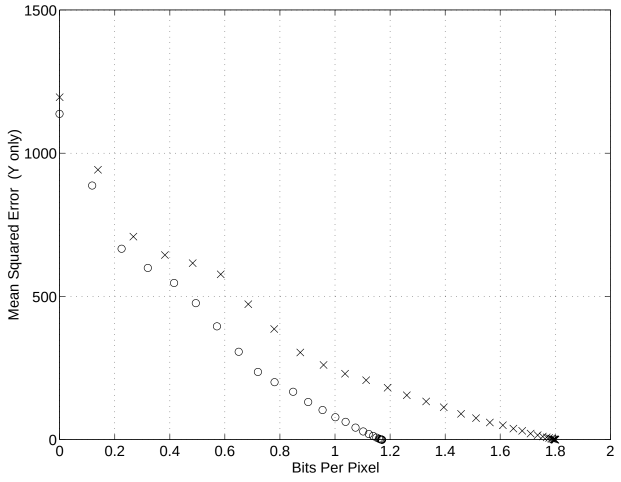


Figure 4: Slice 20 (full-width, frame 0) from "Flower Garden", coded at 24 Mbps (x) and 12 Mbps (o).

The bisection algorithm operates on the convex hull of the $R(D)$ curve of each slice. Consequently, points which lie above that, and hence are not $R(D)$ optimal, are not considered by the algorithm. Figure 4 shows the $R(D)$ plots for an actual slice (frame-based, intra coding of "Flower Garden" at 24 and 12 Mbps, slice 20—full-width—of frame 0). Worth noting is the locally non-convex behavior in both cases. This property can be traced back to the structure of the MPEG-2 run-length encoding tables [2], where specific examples of non-convexity can be easily found. In most cases (and particularly for P and B pictures as we will see later on), the number of $R(D)$ points which lie above the convex hull is small, and hence in practice they do not represent a significant problem.

In some cases, if the $R(D)$ curve of a slice is sufficiently misbehaved, the bisection algorithm can be set off track, with a resulting underutilization of the target bit budget. In order to mitigate this effect, and also to speed up operation, each iteration considers a continuously shrinking interval of possible breakpoint values ("pruning"). This will result in convergence of the algorithm to a much smaller set of non-convex points, and is a byproduct of convexity.

3.3 Performance Evaluation

The collection of necessary data in eq. (14) that is needed to run the algorithm, requires only parsing of the input bitstream up to inverse quantization of the DCT coefficients. In other words, all operations can be performed directly in the compressed domain. Note that distortion data are computed from the luminance component only, whereas rate data are computed from all three components. The parsing process represents a very small fraction of the complete decoding process; the dominant processing step in decoding is in fact the inverse DCT.

The window S (number of consecutive slices) in which the algorithm operates has been considered up to now a design parameter. Since data partitioning is performed after encoding, it is desirable to minimize the additional delay introduced by the extra processing step. Even in cases where partitioning is applied on stored material prior to transmission, delay is a very important parameter for interactive applications. A plausible selection is then a complete single picture (frame or field).

As we mentioned in the previous section, the target bit budget R_{budget} can be set quite arbitrarily, given however that the average rate does not exceed $\hat{B}$. This represents another degree of freedom which is not (and cannot be) optimized by the above algorithm. A convenient selection is obtained by choosing for each picture the value

$$R_{\mathrm{budget}} = (\hat{B}/B)R - R_o \tag{24}$$

where R is the size (in bits) of the currently processed frame (or,more generally, set of slices), and R_o is the number of bits spent for coding components of the bitstream that are not subject to data partitioning (overhead bits for headers, motion vectors, etc.). Allocated bits that are left over from one picture are carried over to the subsequent picture. Note also that R is immediately available after the complete picture has been parsed.

It is very easy to show that the budget selection in (24) guarantees that the target bit rate is not exceeded. We have:

$$R_{\text{budget}} + R_o = \frac{\hat{B}R}{B} \tag{25}$$

and the average value $\bar{R}$ of R over time is $\bar{R} = B$. Hence the average rate for the partitioned picture will be:

$$\overline{R_{\text{budget}} + R_o} = \frac{\hat{B}\bar{R}}{B} = \hat{B} \tag{26}$$

In addition, it is easy to see that this allocation can satisfy any scaled buffering constraints that may be imposed.

The value given by eq. (24) carries over to the partitioning algorithm several properties of the encoder. In particular, we observe that bit allocation is performed in a manner proportional to the one decided by the encoder. Assuming that an "intelligent" encoder has been used, the original bit allocation may have been meticulously optimized; utilizing a proportional allocation in the partitioning process can help to improve the overall video quality. In the case where buffering constraints are imposed to partition 0 for placement prior to transmission, one can convert the problem to a buffer-constrained optimization problem. The approach would be similar to [16], although the problem there was focused on quantizer selection. It is doubtful, however, that the significant extra complexity of the problem can in fact achieve improved results (an optimal fast algorithm for this problem is not known).

The computational overhead of the iterative algorithm is small, with convergence achieved typically within 8–10 iterations. Figure 5 shows the results of applying the algorithm to 20 frames of "Flower Garden", using frame-based intra coding at 24 Mbps, and with a target rate of 12 Mbps for partition 0. The quality metric used is "Y-PSNR", i.e., the Peak SNR of the luminance component only. PSNR is derived from the squared error $e = \|y - p^0\|$ using:

$$\text{PSNR} = -10 \log_{10} \left(\frac{255^2}{e} \right) \quad \text{in dB} \tag{27}$$

where 255 is the peak value for the luminance signal (using an 8-bit representation).

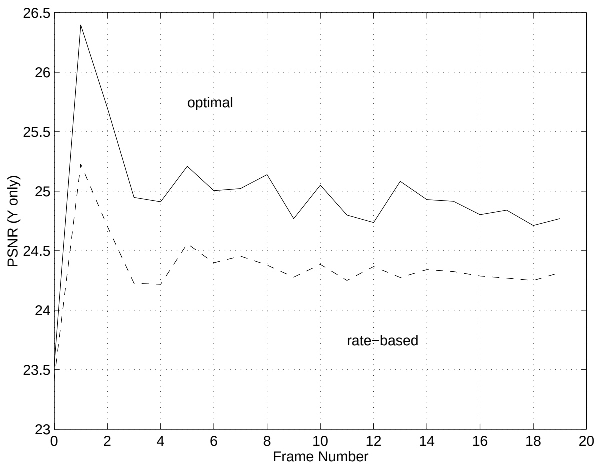


Figure 5: Data partitioning of frame-based, intra coded "Flower Garden", from 24 Mbps to 12 Mbps, using optimal and rate-based algorithms.

Also shown in Figure 5 are the results of a simpler algorithm that uses *rate-based* optimization. In this latter case each slice is independently assigned a target bit budget proportional to its original size R_i, and a breakpoint is selected so that this budget is not exceeded (leftover bits are carried over to the next slice). In other words, we select the breakpoint of each slice as:

$$b_i^* = \max\left\{b_i : R_i(b_i) \leq R_{\mathrm{budget}_i}\right\} \tag{28}$$

The bit budget for each slice is set according to:

$$R_{\mathrm{budget}_i} = \frac{R_i}{R_{\mathrm{budget}}/S} \tag{29}$$

where S is the number of slices. In order to compute R_{budget_i}, a complete picture is read; this makes the algorithms comparable in terms of the optimization window used. Note that this algorithm is purely rate-based, i.e., the distortion is completely ignored. Lagrangian optimization outperforms in this case the rate-based algorithm by 0.6 dB on the average.

4 Data Partitioning in Predictive Coding

4.1 Problem Formulation

When all variants of picture coding types are used (I, P, and B), the problem of data partitioning becomes significantly more complex. The decoding process can be described by:

$$y_i = \mathcal{M}_i(y_{i-1}) + e_i \tag{30}$$

where P_i denotes the i-th decoded picture (in coding order), $\mathcal{M}_i(\cdot)$ denotes the motion compensation operator for picture i, and e_i denotes the coded prediction error. The first picture is assumed to be intra-coded, and hence

$$e_0 = y_0 \tag{31}$$

Although, for simplicity, a single reference picture is shown above for motion compensation, the expression can be trivially extended to cover the general case (which includes B pictures).

By applying data partitioning and decoding partition 0, eq. (30) becomes:

$$\hat{y}_i = \mathcal{M}_i(\hat{y}_{i-1}) + \hat{e}_i \tag{32}$$

where $\hat{e}_i$ denotes the partitioned prediction error. Recall that the original partitioning problem was set in its minimax form in eq. (3) of Section 2.2 (page 8) as follows:

$$\min_{R(p^0)\leq \hat{B}} \left\{ D \overset{\text{def}}{=} \max\left\{ \|y - \hat{y}\| \right\} \right\} \tag{33}$$

Using (30) and (32), and observing that the maximum average distortion is maximized when $\hat{y} = p^0$ (i.e., partition 1 is lost), eq. (33) becomes:

$$\min_{\sum_{i=1}^{N} R_i(b_i)\leq \hat{B}} \left\| \sum_{p=1}^{M} \mathcal{M}_i(y_{i-1}) - \mathcal{M}_i(\hat{y}_{i-1}) + e_i - \hat{e}_i \right\| \tag{34}$$

where M is the number of pictures over which optimization takes place. Note that:

$$\mathcal{M}_i(y_{i-1}) - \mathcal{M}_i(\hat{y}_{i-1}) \neq \mathcal{M}_i(y_{i-1} - \hat{y}_{i-1}) \tag{35}$$

i.e., motion compensation is a non-linear operation, because it involves integer arithmetic with truncation away from zero [2].

From eq. (34) we observe that, in contrast with the intra-only case, optimization involves the accumulated error:

$$a_i \overset{\text{def}}{=} \mathcal{M}_i(y_{i-1}) - \mathcal{M}_i(\hat{y}_{i-1}) \tag{36}$$

Furthermore, due to the error accumulation process, partitioning decisions made for a given picture will have an effect in the quality and partitioning decisions of subsequent pictures. As a result, an optimal algorithm for (34) would have to examine a complete group of pictures (I-to-I), since breakpoint decisions at the initial I-picture may affect even the last B or P picture. Not only the computational overhead would be extremely high, but the delay would be unacceptable as well. It is desirable then to seek fast solutions with small delay, that are able to control error propagation in a well-defined fashion.

An attractive alternative algorithm is one that solves eq. (34) on a picture basis, and where only the error accumulated from past pictures is taken into account. This algorithm will be referred to as *causally optimal.* In addition, in order to avoid similar complications that arise when the optimization window spans more than one picture, we will restrict our discussion for the case where the windows is up to a complete single picture. This property is also an indirect consequence of the causality argument.

Note that in order to accurately compute a_i, two prediction loops have to be maintained: one for a decoder that receives the complete signal, and one for a decoder that receives only partition 0. This is because of the nonlinearity of the integer arithmetic of motion compensation expressed by eq. (35). With the penalty of some lack in arithmetic accuracy, these two loops can be collapsed together. In the following we will assume that the (optimal) dual-loop operation is always used.

4.2 The Causally Optimal Problem

The causally optimal problem can now be formulated as follows. Substituting eq. (36) in (34) we have

$$\min_{R(p^0)\leq \hat{B}} \{\|a_i + e_i - \hat{e}_i\|\} \tag{37}$$

We must now obtain an expression for the total partitioning distortion $a_i + e_i - \hat{e}_i$.

As in the non-predictive case of Section 3.1, we first consider a single block. Let $A(k)$ denote the k-th DCT coefficient of the accumulated error a (in zig-zag scanning order), $E(k)$ the corresponding coefficient of the decoded picture e, $\hat{E}(k)$ the one of the partitioned picture, and b the breakpoint value. We then have

$$\begin{aligned} \|a_i + e_i - \hat{e}_i\| &= \sum_{i=0}^{N-1} \left(A(k) + E(k) - \hat{E}(k)\right)^2 \\ &= \sum_{i=0}^{N-1} A(k)^2 + 2\sum_{i=0}^{N-1} A(k)\left(E(k) - \hat{E}(k)\right) + \sum_{i=0}^{N-1}\left(E(k) - \hat{E}(k)\right)^2 \\ &= \sum_{i=0}^{N-1} A(k)^2 + 2\sum_{i\geq b}^{N-1} A(k)E(k) + \sum_{i\geq b}^{N-1} E(k)^2 \end{aligned} \tag{38}$$

since

$$\hat{E}(k) = \begin{cases} E(k) & \text{if } k < b \\ 0 & \text{otherwise} \end{cases} \tag{39}$$

Observe that the total distortion involves not only the accumulated error and the current picture's partitioning error (identical to the non-predictive case), but crossterms as well.

Due to the independence of individual blocks, we can extend (38) for a complete slice. We should note, however, that while the prediction error DCT coefficients are represented by their run-lengths, and the truncation point is also defined by the number of run-length to be included in partition 0, the accumulated error has no such representation. Consequently, a mapping function $\mathcal{I}(k)$ is necessary that maps the k-th run-length of a block into the appropriate position in the zig-zag scanning pattern.

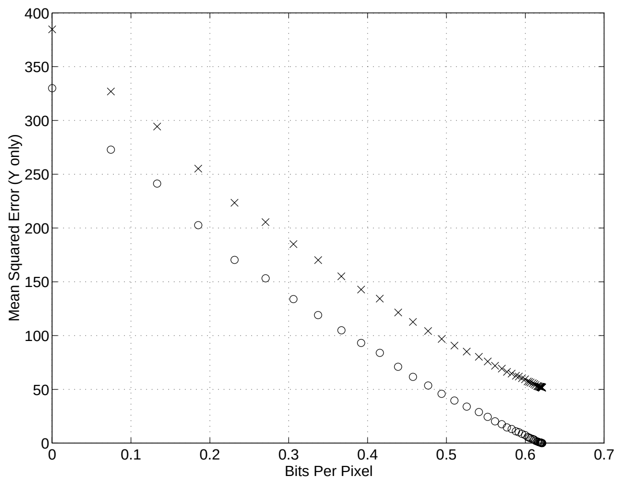


Figure 6: Slice 20 (full-width, frame 3, P-picture) from "Mobile" coded at 4 Mbps and partitioned at 3.2 Mbps: (x) $\hat{D}(B_i)$, (o) $D(B_i)$.

Denoting by $\hat{D}_i(b_i)$ the total partitioning distortion of slice i for the breakpoint value b_i, we have

$$\hat{D}_i(b_i) = \sum_{j \in S_i} \left\{ \sum_{k=0}^{N-1} A_j^i(k)^2 + \sum_{k \geq b_i} 2A_j^i(\mathcal{I}_j^i(k))E_j^i(k) + \sum_{k \geq b_i} E_j^i(k)^2 \right\} \tag{40}$$

where S_i denotes the blocks of slice S_i, $A_j^i(k)$ is the k-th DCT coefficient (in zig-zag scanning order) of the j-th block of the i-th slice of the accumulated error a_i, and $E_j^i(k)$ is the DCT coefficient of the k-th *run-length* of the j-th block of the i-th slice of the coded prediction error. Using (40), the data partitioning problem (37) for the predictive case can be formulated as:

$$\min_{R(p^0) \leq \hat{B}} \{\|a_i + e_i - \hat{e}_i\|\} \Longleftrightarrow \min_{\sum_{i=1}^{N} R_i(b_i) \leq \hat{B}} \left\{ \sum_{i=1}^{S} \hat{D}_i(b_i) \right\} \tag{41}$$

where S is the total number of slices in the optimization window.

4.3 The Causally Optimal Algorithm

The minimization problem in (41) can be solved using the Lagrangian optimization approach of the non-predictive case in Section 3.2, using the more general definition of the distortion $\hat{D}$ given by eq. (40).

Of particular concern is the convexity of the $R(D)$ curves when the total distortion (including the accumulated error) is taken into account. Figure 6 shows the R(D) curve for slice 20 of frame 3 (P-picture) from the sequence "Mobile" coded at 4 Mbps (frame-based coding) and partitioned at 3.2 Mbps using the causally optimal algorithm. The upper curve takes into account the accumulated error a_i, whereas the bottom one involves only the prediction error partitioning distortion $e_i - \hat{e}_i$.

We observe that convexity is clearly present. In fact, for predicted pictures, $R(D)$ curves tend to be uniformly convex, compared with intra pictures which tend to have a concave middle segment. We have experimentally verified that this property holds even for small slice sizes (e.g., 11 or 4 macroblocks per slice, instead of the regular 44 which amounts to the whole picture width), although the curves become progressively flatter.

4.4 Performance Evaluation

An important issue in mixed-mode coding, as in non-predictive coding, is the target bit budget that will be set for each picture (or more generally, set of slices). The matter is more complicated than in the intra-only case, due to the irregular bit distribution among pictures of different types. In a typical situation, I and P picture DCT coding requires a significant number of bits, while B picture sizes are dominated by header and motion vector coding bits. Figure 7 depicts the number of total vs. overhead bits for the "Mobile" sequence coded at 4 Mbps. "Overhead" here includes everything except the DCT coefficients which are subject to partitioning. Observe the evident periodic pattern between I pictures, and the irregularity of the bit distribution between I, P, and B pictures.

As a result of the bit distribution irregularity, B pictures provide much less flexibility for data partitioning. In order to accommodate this behavior, I and P pictures are assigned proportional bit budgets as in Section 3.3. For B pictures the same is done, except when the resulting bit budget is negative, in which case it is set to 0. The negative budget, however, is accounted for, so

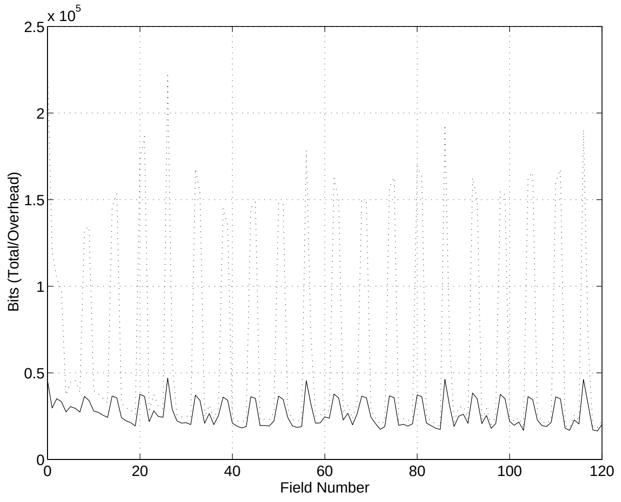


Figure 7: Bit distribution for the "Mobile" sequence coded at 4 Mbps, with I period 12, and B period 3 (the overhead bits include all non-DCT bitstream components).

that the bits spent for the B picture are subtracted from the budget of the immediately following picture. Note that an optimal bit allocation for each picture would be a direct by-product of the optimal non-causal algorithm.

Figure 8 shows the Y-PSNR resulting from the causally optimal algorithm on 15 frames of the "Mobile" sequence (I distance N=15, I/P frame distance M=3), frame-based coded at 4 Mbps and partitioned at 3.2 Mbps (80% of the rate goes to partition 0). This is the signal quality that would be observed by a decoder that receives only partition 0, compared with one that receives both partitions. We see that I and P frames suffer the most, while B frames are in general up to 1 dB better.

The complexity of solving eq. (41) is significant, as it requires at least one complete decoding loop for the luminance signal. If the non-linearity of the motion compensation is taken into account, then two such loops are required. In addition, since motion compensation is performed in the spatial domain while partitioning is performed in the DCT domain, a forward DCT computation module is required as well in order to compute $A_j^i(\cdot)$. As a result, the implementation complexity is between that of a decoder and an encoder.

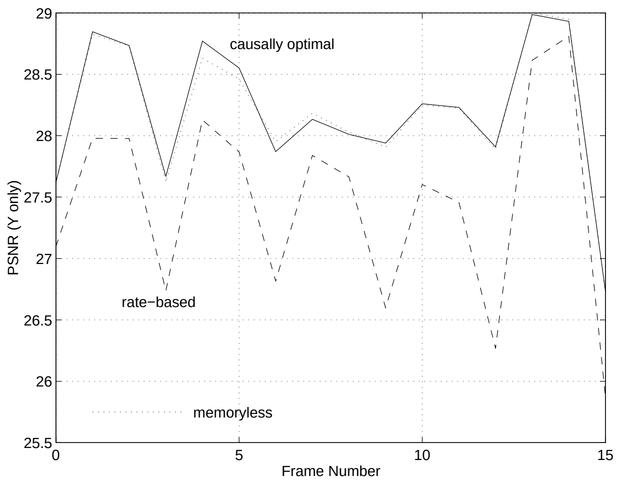


Figure 8: PSNR (Y only) for "Mobile" sequence, frame-based coded at 4 Mbps and partitioned at 3.2 Mbps using the causally optimal, memoryless, and rate-based algorithms.

4.5 The Memoryless and Rate-Based Algorithms

Given the complexity of the causally optimal algorithm, it is interesting to examine the benefit of error accumulation tracking. This can be evaluated by applying the algorithm of Section 3.2 (intra-only case) to the mixed-mode case, since the only difference is the accumulated error term a_i. Bit budget allocation, however, is performed as discussed in Section 4.4 (mixed-mode case).

Surprisingly, the results of this *memoryless* mixed-mode partitioning algorithm are almost identical to the causally optimal one. Figure 8 shows the relevant PSNR values for the "Mobile" sequence. The difference is in general less than 0.1 dB and the curves can hardly be distinguished. We have experimentally verified that this holds for a very wide range of bit rates (i.e., down to 50% reduction, or more depending on the original rate and picture resolution) and slice sizes. The difference, however, increases slightly to 0.2–0.3 dB. We should note that these difference values are perceptually insignificant.

This is a very important result, as it implies that we can dispense completely with the error accumulation calculation and its associated computational complexity, for a minimal cost in performance. The quality degradation between the causally optimal and memoryless algorithms will be perceptually insignificant, across the spectrum of slice sizes and partition rates.

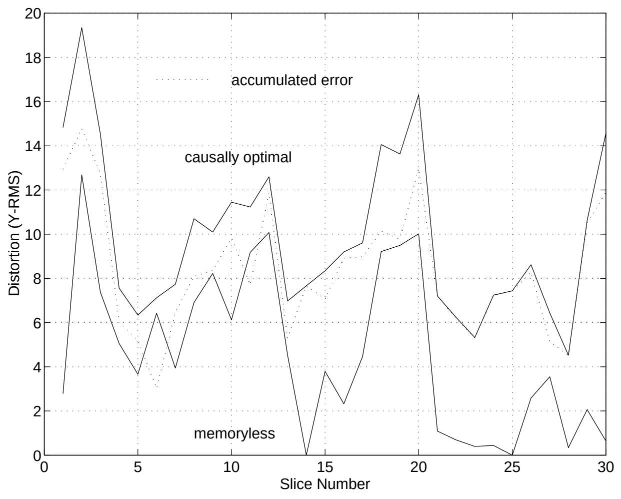


Figure 9: Distribution of accumulated error, causally optimal, and memoryless distortions across all slices of a picture ("Mobile", coded at 4 Mbps and partitioned at 3.2 Mbps, frame 3, P picture).

This property is hinted at by Figure 6 upon closer examination. The upper and lower curves are almost identical, except for a vertical shift. Figure 9 depicts the two types of distortions (from the causally optimal and memoryless algorithms) as well as the accumulated error across all slices of a picture. We observe that the two distortions behave in very similar ways as we move along the picture. In order for the accumulated error to affect the partitioning decisions, either the slope of the $R(D)$ curves or the overall accumulated error distribution across a picture would have to be significantly affected. This, however, is not the case, because at each picture the partitioning decisions are optimally made.

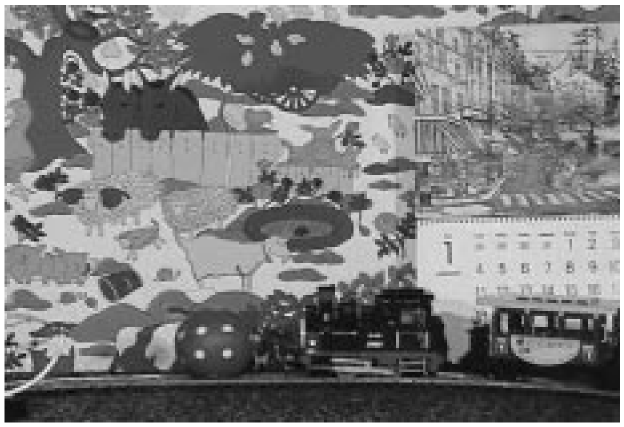
Finally, we examine the performance of the rate-based optimization algorithm introduced in Section 3.3 (eq. (28)), in a mixed-mode coding environment. Since, as was previously pointed out, rate-based optimization does not take into account the distortion, there is no difference whether or not the accumulated error is tracked. Consequently, the only difference lies in the bit budget allocation. Figure 8 depicts the results obtained on the "Mobile" sequence, with the same coding and partitioning parameters as before. We see that the rate-based algorithm is inferior by about 1 dB. The complexity, however, is significantly reduced as well, as the Lagrangian optimization iteration is avoided. This makes the rate-based algorithm attractive, when complexity and/or implementation costs are of importance.

Figure 10 shows the same reconstructed frame (luminance only) using the various partitioning algorithms[4]. The frame is what a decoder would display if only partition 0 was received, except from Figure 10(a) which is the decoded frame at full rate. The figures show frame number 12 (a P picture) from the "Mobile" sequence, coded at 4 Mbps and partitioned at 3.2 Mbps. We can see instances where the optimal (causally and memoryless) algorithms perform better than the rate-based scheme, but there are a few cases where the reverse is true as well. For example, the small birds in the middle of the top-most slices are better in the optimal algorithms; the same is true for the dotted ball (bottom left), the border of the stream engine and the car, as well as the 3rd row of the calendar's numbers (11–17). There are two cases, however, where the allocation performed by the rate-based scheme provides better perceptual results: the duck in the middle of the left-hand side border is sharper, as well as the 2nd row of the calendear's numbers (4–10). These results corroborate our SNR evaluations, indicating that the rate-based approach, although definetely inferior, is still a competitive technique.

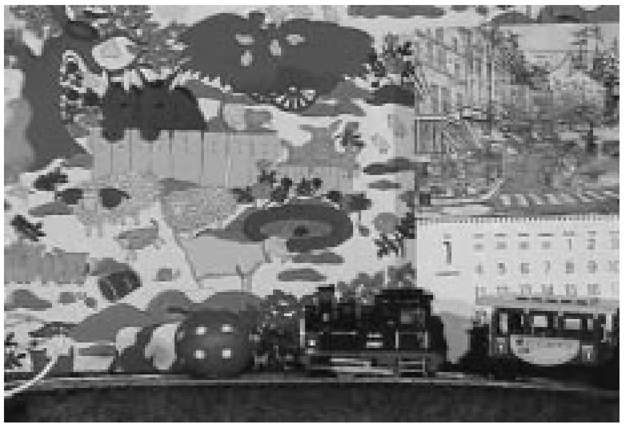
5 Concluding Remarks

The problem of optimal data partitioning in motion-compensated transform coding was analyzed, with particular emphasis in its use by the MPEG-2 video coding standard. Data partitioning can be a very effective tool for transmission of single-layer video bitstreams over unreliable channels, including channels that provide prioritized transmission (i.e., virtual channels in packet-based networks). A key property of the approach is that it can be applied even after encoding has already taken place, and thus is applicable not only for live transmission systems, but also for stored video applications such as video-on-demand. A potential drawback of the approach is that, in contrast with other scalable coding approaches, neither of the two partitions in which the bitstream is split

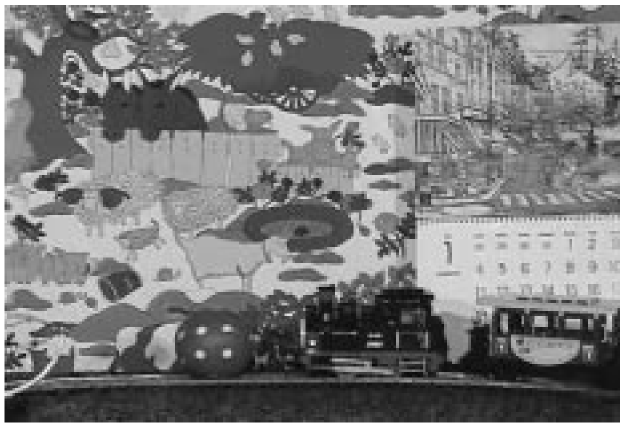
[4]The images here hare halftoned and scaled down for printing. The original images can be accessed at `http://www.ee.columbia.edu/~eleft/dp` (in PGM format, luminance only).



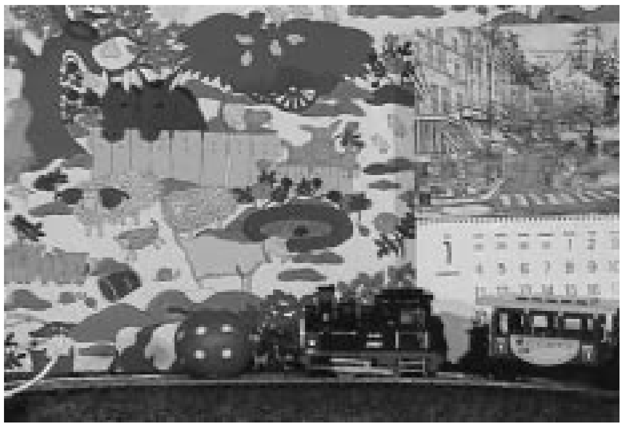
(a) No partitioning



(b) Causally optimal partitioning



(c) Memoryless optimal partitioning



(d) Rate-based partitioning

Figure 10: Reconstructed frame using various data partitioning algorithms ("Mobile", coded at 4 Mbps and partitioned at 3.2 Mbps, frame 12, P picture).

is self-contained. Consequently, due to the recursive nature of motion-compensated compression, if part of the bitstream is lost (namely, from partition 1), error accumulation will occur.

We provided an analysis of data partitioning in an operational rate-distortion context. An optimal algorithm based on Lagrange multipliers was derived for non-predictive (intra-only) coding, and shown to be less complex than a full decoder. For the predictive coding, or mixed-mode case (I, P, and B pictures) the optimal algorithm was characterized and shown to possess significantly high complexity and delay, as a complete group of pictures was required to be processed at a time. As an alternative, a "causal" minimization formulation was proposed, in which only the accumulated error from past pictures is taken into account (while the one propagated to future pictures is ignored).

An optimal algorithm for the causal problem was developed as a generalization of the non-predictive case. Experimental results have shown that the algorithm performs quite well, with P and B pictures having about 1 dB higher quality than I ones. It was then shown that tracking the error accumulation from one frame to the next does not actually benefit the partitioning process in any significant way, and hence that a memoryless algorithm employing Lagrangian optimization is sufficient.

Finally, a faster but suboptimal algorithm that uses rate-based optimization was also proposed for comparison purposes. It was shown to perform quite close (within 0.6 dB on the average) to the optimal one for the intra-only case, but proved to be inferior by 1 dB on the average for the mixed-mode case. Nevertheless, its simplicity makes it potentially attractive for low cost implementations.

References

[1] Information Technology – Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to About 1,5 Mbit/s, ISO/IEC International Standard 11172 (MPEG-1). 1993.

[2] Information Technology – Generic Coding of Moving Pictures and Associated Audio, ITU-T Recommendation H.262, ISO/IEC International Standard 13818 (MPEG-2). 1996.

[3] T. Berger. *Rate Distortion Theory.* Prentice Hall, Englewood Cliffs, New Jersey, 1971.

[4] T. Cover and J. Thomas. *Elements of Information Theory.* John Wiley & Sons, New York, New York, 1991.

[5] A. Eleftheriadis. *Dynamic Rate Shaping of Compressed Digital Video.* PhD thesis, Columbia University, New York, New York, 1995.

[6] A. Eleftheriadis and D. Anastassiou. Optimal Data Partitioning of MPEG-2 Coded Video. In *Proceedings, 1st IEEE International Conference on Image Processing*, pages I.273–I.277, Austin, Texas, November 1994.

[7] H. Everett. Generalized Lagrange Multiplier Method for Solving Problems of Optimum Allocation of Resources. *Operations Research*, 11:399–417, 1963.

[8] A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, Boston, Massachusetts, 1992.

[9] B. G. Haskell, A. Puri, and A. N. Netravali. *Digital Video: An Introduction to MPEG-2.* Chapman and Hall, New York, 1997.

[10] N. S. Jayant and P. Noll. *Digital Coding of Waveforms: Principles and Applications to Speech and Video.* Prentice Hall, Englewood Cliffs, New Jersey, 1984.

[11] K. Ramchandran and M. Vetterli. Rate-distortion optimal fast thresholding with complete JPEG/MPEG decoder compatibility. In *Proceedings, Picture Coding Symposium '93*, March 1993.

[12] K. Ramchandran and M. Vetterli. Rate-distortion optimal fast thresholding with complete JPEG/MPEG decoder compatibility. *IEEE Transactions on Image Processing, Special Issue on Video Sequence Compression*, 3(5):700–704, September 1994.

[13] Didier LeGall. MPEG: A Video Compression Standard for Multimedia Applications. *Communications of the ACM*, 34(4):46–58, April 1991.

[14] J. L. Mitchell, W. B. Pennebaker, D. LeGall, and C. Fogg. *MPEG Video Compression Standard.* Chapman and Hall, New York, 1997.

[15] A. N. Netravali and B. G. Haskell. *Digital Pictures: Representation, Compression, and Standards (2nd ed.).* Plenum Press, New York, New York, 1995.

[16] A. Ortega, K. Ramchandran, and M. Vetterli. Optimal Buffer-Contrained Source Quantization and Fast Approximations. In *Proceedings, IEEE Intl. Symposium on Circuits and Systems, ISCAS '92*, San Diego, May 1992.

[17] A. Reibman. DCT-Based Embedded Coding of Packet Video. *Image Communication*, 3:231–237, 1991.

[18] Y. Shoham and A. Gersho. Efficient bit allocation for an arbitrary set of quantizers. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(9):1445–1453, 1988.