Katsaggelos, Aggelos K. and Gerry Melnikov “Rate-Distortion Techniques in Image and Video Coding”
*Multimedia Image and Video Processing*
Ed. Ling Guan et al.
Boca Raton: CRC Press LLC, 2001
Chapter 11

Rate-Distortion Techniques in Image and Video Coding

Aggelos K. Katsaggelos and Gerry Melnikov

11.1 The Multimedia Transmission Problem

One of the central issues in multimedia communications is the efficient transmission of information from source to destination. This broad definition includes a large number of target applications. They may differ in the type (i.e., text, voice, image, video, etc.) and form (analog or digital) of the data being transmitted. Applications also differ by whether lossless or lossy data transmission is employed and by the kind of channel the transmitted data passes through on its way from an encoder to a decoder or a storage device. Additionally, there are wide variations in what represents acceptable loss of data. For example, whereas certain loss of information is acceptable in compressing video signals, it is not acceptable in compressing text signals.

In general, the goal of a communications system is the transfer of data in a way that is resource efficient, minimally prone to errors, and without significant delay. In a particular application we may know the target operating environment (bit rate, channel characteristics, acceptable delay, etc.). The challenge is to arrive at the data representation satisfying these constraints. For example, if the application at hand is transmission of digital video over a noisy wireless channel, the task is to allocate the available bits within and across frames optimally based on the partially known source statistics, the assumed channel model, and the desired signal fidelity at the receiver.

In problems where some information loss is inevitable, whether caused by quantizing a deterministic signal or through channel errors, rate distortion theory provides some theoretical foundations for deriving performance bounds. The fundamental issue in rate distortion theory is finding the bound for the fidelity of representation of a source of known statistical properties under a given description length constraint. The dual problem is that in which a given fidelity requirement is prescribed and the source description length is to be minimized.

Although there is no argument that the count of bits should be the metric for rate or the description length, the choice of the corresponding metric for distortion is not straightforward and may depend on the particular application. The set of all possible distortion metrics can be broadly categorized into unquantifiable perceptual metrics and metrics for which a closed-form mathematical representation exists. The latter class, particularly the mean squared error (MSE) metric, and, to a lesser extent, the metric based on the maximum operator [34], has been primarily used in the image and video coding community partly due to the ease of computation and partly for historical reasons.
The well-known Shannon’s source and channel coding theorems [35] paved the way for separating, without loss of optimality, the processes of removing redundancy from data (source coding) and the transmission of the resulting bitstream over a noisy channel (channel coding). Shannon also established a lower (entropy) bound on the performance of lossless source compression algorithms and subsequent error-free transmission over a channel of limited capacity. Hence, most practical communication systems have the structure shown in Figure 11.1.

![Figure 11.1](image)

**FIGURE 11.1**
High-level view of a video communications system.

Many applications demand data compression beyond entropy (i.e., with some loss of information). Wireless transmission of video is one example when the original data stream must be compressed by factors of hundreds or thousands, due to limited channel capacity. Clearly, with lossy compression, there is a trade-off between the transmission of fewer bits through a channel and the quality of the reconstructed signal. The central issue in the rate-distortion theory is how much redundancy can be removed from a given data source while satisfying a distortion constraint. Conversely, the constraint can be imposed on the rate, in which case a lower bound on distortion is sought. The theory is based on knowledge of the source statistics and not on any specific source coding scheme. Hence it provides a lower bound on the rate, which clearly may not be achieved by a particular encoder implementation. In addition to not being constructive, this bound assumes perfect knowledge of the source statistics which, in most cases, is not available. Source model complexity is yet another dimension in rate-distortion theory. The trade-off here is between a very simple source model (e.g., an i.i.d. Gaussian model), which is relatively easy to treat analytically, on one hand, and a model that is better adapted to a given source but is more complex. Clearly, any performance bounds obtained with the help of the rate-distortion theory are only as good as the source models and may be of little use if they are inaccurate.

Even though, traditionally, rate-distortion theory was applied only to situations where distortion was caused deterministically through quantization at the encoder, it is applied equally well to the case when the distortion is introduced stochastically in a noisy channel. In this case, however, the tightness of the bound also depends on the accuracy of the channel model.

The applicability of Shannon’s separation principle between source and channel coding for video transmission, under some real-world constraints, was studied in [20] and the references therein. The conclusion is that for applications such as the Internet, with its packetization and delay constraints, for multicast and layered applications, where the decoding must be possible for any subset of the received bitstream, as well as for applications requiring unequal error protection, utilizing the separation principle yields results inferior to those obtained by joint source–channel coding. Practical application of Shannon’s bound is also hampered by such factors as the infinite block size (symbol length) assumption, and the corresponding time delay.
As discussed in Section 11.3, some of these real-world constraints can be easily incorporated into a rate-distortion optimized coding algorithm.

Lossless data compression techniques capitalize on redundancies present in the source and do not involve any trade-offs between rate and distortion. When it comes to designing lossy data compression algorithms, however, the challenge is to represent the original data in such a way that the application of coarse quantizers, which is what makes compression beyond entropy possible, costs the least in terms of the resulting degradation in quality.

The objective of rate-distortion theory is to find a lower bound on the rate such that a known source is represented with a given fidelity, or conversely. Thus, if the rate \( R \) and distortion \( D \) are quantified, varying the maximum distortion or fidelity continuously, and finding the corresponding bound \( R \), is equivalent to tracing a curve in the \( R-D \) space. This curve is called the rate-distortion function (RDF). Even assuming that the source model is correct, points on this curve, however, may be difficult to achieve not only because of the infinite block length and computational resources assumption, but also because they represent a lower bound among all possible encoders under the sun.

### 11.2 The Operational Rate-Distortion Function

In the previous section we discussed the classical rate-distortion theory, the primary purpose of which is to establish performance bounds for a given data source across all possible encoders, operating at their “best.” In practice, however, we often deal with one fixed encoder structure and try to optimize its free parameters. Finding performance bounds in such a setup is the subject matter of the operational rate-distortion (ORD) theory.

Operating within the framework of a fixed encoder simplifies the problem a lot. Solving the problem of optimal bit allocation in this restricted case, however, does not guarantee operation at or near the bound established by the RDF. As an example, let us consider the simplest image encoder, which approximates the image by the DC values of blocks derived from a fixed partition. Clearly, finding the best possible quantization scheme for these coefficients, while making the encoder optimal, will not do a good job of compressing the image or come close to the RDF, since the encoder structure itself is very simple.

ORD theory is based on the fact that every encoder maps the input data into independent or dependent sources of information which need to be represented efficiently. The finite number of modes the encoder can select for each of these data subsources can be thought of as the admissible set of quantizers. That is, a particular quantization choice for each of the subsources in the image or video signal, for example, constitutes one admissible quantizer. A quantizer, in this case, is defined in the most general terms. It may operate on image pixels, blocks, or coefficients in a transformed domain. The encoder, in addition to being responsible for selecting quantizers (reconstruction levels) for each of the subsources, must arrive at a good partition of the original signal into subsources. Let us call a particular assignment of quantizers to all subsources a \textit{mode of operation} at the encoder. Each mode can be categorized by a rate \( R \) and a distortion \( D \). The set of all \( (R, D) \) pairs, whose cardinality is equal to the number of admissible modes, constitutes the quantizer function (QF). Figure 11.2, in which the QF is shown with the \( \times \) symbol, illustrates this concept.

The ORD curve, denoted by the dotted line in Figure 11.2, is a subset of the QF and represents modes of desirable operation of the encoder. Mathematically, the set of points on the ORD curve is defined as follows:

\[
\text{ORD} = \{ i \in \text{QF} : R_i \leq R_j \text{ or } D_i \leq D_j, \forall j \in \text{QF}, i \neq j \},
\]

\( \text{(11.1)} \)
where $R_i, D_i$ and $R_j, D_j$ are the rate-distortion pairs associated with modes $i$ and $j$, respectively. For convenience, it is customary to connect consecutive points of the ORD set, thus forming the ORD curve. Clearly, it is desirable to operate on the ORD curve, and modes not on the curve are not optimal in the sense that within the same encoder, a smaller distortion is achievable for the same or smaller rate, or vice versa.

As stated, the concept of operating on the ORD curve is quite generic and can be applied to a variety of applications. Practically, the idea of rate-distortion optimization is equivalent to bit budget allocation among different subsources of information in a given compression or channel transmission framework. When an allocation scheme results in a mode belonging to the ORD set, an algorithm is operating at its optimum.

The framework of bit allocation with the goal of reaching a point on the ORD curve applies equally well in cases when the source is stochastic in nature and is known only through the model of its probability density function ($pdf$), or in cases when transmission through a noisy channel, rather than compression, is the problem. In these cases, $E(D)$, the expected distortion, is used instead of distortion, and the tightness of the bound established by the ORD curve is sensitive to source or channel model accuracy.

### 11.3 Problem Formulation

The central problem in ORD optimization is to select appropriately the modes of the given algorithm such that a point on the ORD curve is reached. This is equivalent to saying that no other selection of parameters would lead to a better distortion performance for the same bit rate. Following the notation of [28], if $B$ is a code belonging to the set of all possible modes $S_B$ generated by the given algorithm as a code for the specified data source, and $R(\cdot)$ and $D(\cdot)$
are the associated rate and distortion functions, we seek a mode \( B^* \), which is the solution to the following constrained optimization problem,

\[
\min_{B \in S_B} D(B), \quad \text{subject to: } R(B) \leq R_{\max}.
\]

(11.2)

It turns out that the problem dual to that of (11.2), in which the source fidelity or distortion is constrained, can be solved using the same tools (described in Section 11.4) as the rate-constrained problem. The dual problem can be expressed as follows:

\[
\min_{B \in S_B} R(B), \quad \text{subject to: } D(B) \leq D_{\max}.
\]

(11.3)

As stated, these problems are general enough to include many possible distortion metrics and many parameter encoding schemes, including differential ones. Since the concept of rate and distortion is typically associated with quantizers, it often helps to think of solutions to (11.2) and (11.3) as optimal bit allocations among (possibly dependent) quantizers.

### 11.4 Mathematical Tools in RD Optimization

In this section we discuss two powerful optimization methods: the Lagrangian multiplier method and dynamic programming (DP). These techniques are very suitable for the kind of problems discussed here, that is, the allocation of resources among a finite number of dependent quantizers. For an overview of optimization theory the reader is referred to [24]. In all examples presented in this chapter, these two tools are used in conjunction with each other. First, the Lagrangian multiplier method is used to convert a constrained optimization problem into an unconstrained one. Then, the optimal solution is found by subdividing the whole problem into parts with the help of DP.

#### 11.4.1 Lagrangian Optimization

The Lagrangian multiplier method described here is the tool used to solve constrained optimization problems. The idea behind the approach is to transfer one or more constraints into the objective function to be minimized. In the context of image and video coding, the most commonly used objective function is the distortion, with the bit rate being the constraint. As stated, this problem is difficult because it provides no quantifiable measure by which an encoder can make a local decision of selecting the best quantizer among several available quantizers. The Lagrangian multiplier method solves this problem by adding the rate constraint to the objective function, thereby redefining it.

Mathematically, this idea can be stated as follows. Finding the optimal solution \( B^*(\lambda) \) of

\[
\min_{B \in S_B} (D(B) + \lambda \cdot R(B)),
\]

(11.4)

where \( \lambda \) is a positive real number, is equivalent to solving the following constrained optimization problem:

\[
\min_{B \in S_B} D(B), \quad \text{subject to: } R(B) \leq R_{\max}.
\]

(11.5)

Clearly, the optimal solution \( B^* \) is a function of \( \lambda \), the Lagrangian multiplier. It is worth noting that the converse is not always true. Not every solution to the constrained problem can
be found with the unconstrained formulation. That is, there may be values of $R_{\text{max}}$, achievable
optimally by exhaustive search or some other method, for which the corresponding $\lambda$ does not
exist and, therefore, is not achievable by minimizing (11.4).

Since, in practice, the constrained problem needs to be solved for a given $R_{\text{max}}$, a critical
step in this method is to select $\lambda$ appropriately, so that $R(B^*(\lambda)) \approx R_{\text{max}}$. Choosing such a $\lambda$
can be thought of as determining the appropriate trade-off between the rate and the distortion,
which is application specific.

A graphical relationship between points on the ORD curve and line segments in the first
quadrant with slope $-\frac{1}{\lambda}$ can be established, based on the fact that the rate and distortion
components of points on the ORD curve are a nonincreasing and a nondecreasing function of
$\lambda$, respectively [26, 28]. That is, as Figure 11.3 shows, if we start with a line of that slope
passing through the origin and keep moving in the northeast direction, the sweeping line will
first intersect the ORD curve at the point(s) corresponding to the rate and the distortion that
are the optimal solutions to (11.4) when the trade-off of $\lambda$ is used.

![Graphical relationship between points on the ORD curve and line segments](image)

**FIGURE 11.3**
The line of slope $-\frac{1}{\lambda}$ intersects the ORD curve at points having the rate-distortion trade-off of $\lambda$.

There exist two fundamentally different approaches to finding $\lambda_{\text{optimal}}$ meeting the $R_{\text{max}}$
budget. The first approach assumes a continuous model $D(R)$ for the ORD function, for
example, a decaying exponential. Then, $\lambda_{\text{optimal}}$ is approximated through $\lambda = -\frac{dD}{dR}$ evaluated
at $R = R_{\text{max}}$. A model-based technique, however, is only as good as the model itself.
The second approach, which is also based on the monotonicity of \( R(\lambda) \) and \( D(\lambda) \), uses an iterative search to find \( \lambda_{\text{optimal}} \). The process consists of running the encoder for two different values of \( \lambda, \lambda_l \) and \( \lambda_u \), corresponding to the beginning and the end of the interval to which \( \lambda_{\text{optimal}} \) belongs. This interval is iteratively redefined either with the bisection method or based on a fast Bezier curve search technique [28].

### 11.4.2 Dynamic Programming

Dynamic programming (DP) is a tool that is typically applied to optimization problems in which the optimal solution involves a finite number of decisions, and after one decision is made, problems of the same form, but of a smaller size, arise [2]. It is based on the principle that the optimal solution to the overall problem consists of optimal solutions to its subparts or subproblems. In problems of this type exhaustive search solves the same subproblems over and over as it tries to find the global solution at once. By contrast, DP solves each subproblem just once and their solutions are stored in memory.

In the image and video coding applications considered in this chapter, optimality is achieved by finding the ordered sequence of quantizers \([q_0^*, \ldots, q_N^*]\), minimizing the overall cost function,

\[
J^* (q_0^*, \ldots, q_N^*) = \min_{q_0, \ldots, q_N} J (q_0, \ldots, q_N),
\]

where both the quantizers and their number, \( N \), have to be determined by the encoder. In the context of rate- or distortion-constrained optimization, DP is combined with the Lagrangian multiplier method, in which case the cost function is often written as \( J_\lambda \) to emphasize its dependence on \( \lambda \). The DP method is applicable because the total cost function minimization can be broken down into subproblems in a recursive manner as follows:

\[
J^* (q_0^*, \ldots, q_N^*) = \min_i \left[ J^* (q_0^*, \ldots, q_i^*) + J^* (q_{i+1}^*, \ldots, q_N^*) \right], \quad 0 \leq i \leq N.
\]

It should be noted, however, that straightforward application of DP to problems of very large dimensions may be impractical due to coding delay considerations. Even though DP results in solutions of significantly lower complexity than exhaustive search, it nevertheless performs the equivalent of exhaustive search on the local level. In such cases, a greedy suboptimal matching pursuit approach, based on incremental return, may be called for. This was done in [18] and [6] in the context of low-bit-rate video and fractal compression, respectively.

### 11.5 Applications of RD Methods

In this section we describe the application of RD-based methods to several different areas of image and video processing. These include motion estimation, motion-compensated interpolation, object shape coding, fractal image compression, and quad-tree (QT)-based video coding. The common theme shared by these applications is that they all can be stated as resource allocation problems among dependent quantizers (i.e., in the forms shown in Section 11.3). Operationally optimal solutions are obtained in each case and are shown to significantly outperform traditional heuristic approaches. In all these applications the mathematical tools described in Section 11.4 are used in the optimization process.
11.5.1 QT-Based Motion Estimation and Motion-Compensated Interpolation

Most of the resources in present-day multimedia communication systems are devoted to
digital video, because its three-dimensional nature is inherently more complex than that of
speech signals and text. The function of a video codec is unique for several reasons. First, the
large amount of raw data necessitates high compression efﬁciency. Second, a typical video
waveform exhibits correlation in the spatial and, to a greater extent, in the temporal direction.
Furthermore, the human visual system is more sensitive to errors in the temporal direction.

Motion compensation (MC) is the technique most commonly used in video coding to capi-
talize on temporal correlation. In this context, some frames in a video sequence are encoded
with still image coding techniques. These frames are called I frames. The second type of
frames, P frames, are predicted from their reference frames using motion information, which
has to be estimated, usually on a block-by-block basis. A set of motion vectors, deﬁning for
each pixel in the current frame its location in the reference frame, is called the displacement
vector ﬁeld (DVF).

Due to its simplicity and easy hardware implementation, block matching is the most popular
method of motion estimation. Although the use of irregularly shaped regions potentially allows
for better local adaptivity of the DVF to a frame’s spatial segmentation, its application in video
coding is hampered by the need to transmit shape information. By allowing segmentation into
blocks of variable sizes, a compromise between a compact representation (blocks of a ﬁxed
size) of the DVF and its local adaptivity (complex object-oriented approach) is achieved. That
is, large blocks can be used for the background and smaller blocks can be used for areas in
motion. The QT structure is an efﬁcient way of segmenting frames into blocks of different
sizes, and it was used to represent the inhomogeneous DVF in [11, 28]. In addition, the
QT approach enables a tractable search for the joint and optimal segmentation and motion
estimation, which is not possible with the complex object-based segmentation [28].

Optimal Motion Estimation

In this section an operationally optimal motion estimator is derived [28, 32]. The overall
problem solved here can be stated as that of minimizing the displaced frame difference (DFD)
for a given maximum bit rate, and with respect to a given intra-coded reference frame. It
should be noted, however, that some efforts have been made to jointly optimize the anchor
and the motion-compensated frame encoding [10]. The QT-based frame decomposition used
here is achieved by recursively subdividing a $2^N \times 2^N$ image into subimages of halved size
until the block size $2^{n_0} \times 2^{n_0}$ is reached. This decomposition results in an $(N - n_0 + 1)$-level
hierarchy, where blocks at the $n$th level are of size $2^n \times 2^n$.

Within this decomposition, each square block $b_{l,i}$ ($l$th level in the QT, $i$th block in that level)
is associated with $M_{l,i}$ — the set of all admissible motion vectors, of which $m_{l,i}$ is a member.
Then a local state $s_{l,i} = [l, i, m_{l,i}] \in S_{l,i} = [l] \times [i] \times M_{l,i}$ can be deﬁned for each block
$b_{l,i}$. Consequently, a global state $x$, representing the currently chosen local state, is deﬁned as
$x \in X = \bigcup_{l=0}^{N} \bigcup_{i=0}^{2^{n_0} - 1} S_{l,i}$, where $X$ is the set of all admissible global state values.

A complete description of the DVF in the rate-distortion framework requires that the in-
dividual block states $s_{l,i}$ be enumerated sequentially because the rate function may involve
arbitrary-order dependencies resulting from differential encoding of parameters. Hence, the
code for a predicted frame consists of a global state sequence $x_0, \ldots, x_{N_0-1}$, which represents
the left-to-right ordered leaves of a valid QT $\Theta$.

In this context, the frame distortion is an algebraic sum of block distortions $d(x_j)$, where
the blocks correspond to the leaves of the chosen QT decomposition $\Theta$. That is,
\[ D \left( x_0, \ldots, x_{N_{\Theta L} - 1} \right) = \sum_{j=0}^{N_{\Theta L} - 1} d \left( x_j \right). \] (11.8)

The individual block distortion metric chosen here is the MSE of the DFD projected on the block.

Encoding the motion vectors of the DVF is a challenging task in itself. On the one hand, high coding efficiency can be achieved by considering long codewords composed of many motion vectors \( m_{l,j} \) along the scanning path. On the other hand, as explained in Section 11.4, the complexity of the optimal solution search is directly related to the order of dependency in the differential encoding of parameters. As a compromise between the two, a first-order DPCM scheme is used, allowing a first-order dependency between the leaves along the scanning path.

Another challenge is posed by the fact that a typical image exhibits intensity and motion vector correlation in two dimensions, whereas a scanning path is inherently one dimensional. A scan according to the Hilbert curve was shown in [28, 32] to possess certain space-filling properties and create a representation of the 2D data, which is more correlated than that resulting from a raster scan. It can be generated in a recursive fashion and is natural for QT-decomposed images.

Based on the chosen first-order DPCM along a Hilbert scanning path, the overall frame rate can be expressed as follows:

\[ R \left( x_0, \ldots, x_{N_{\Theta L} - 1} \right) = \sum_{j=0}^{N_{\Theta L} - 1} r \left( x_{j-1}, x_j \right), \] (11.9)

where \( r \left( x_{j-1}, x_j \right) \) is the block bit rate, which is a function of the quantizers used for encoding the current and the previous blocks.

Having defined the distortion and the rate, the problem of motion estimation is posed as a constrained optimization problem as follows:

\[ \min_{x_0, \ldots, x_{N_{\Theta L} - 1}} D \left( x_0, \ldots, x_{N_{\Theta L} - 1} \right), \quad \text{subject to:} \quad R \left( x_0, \ldots, x_{N_{\Theta L} - 1} \right) \leq R_{\text{max}}. \] (11.10)

Since the process of approximating a block \( b_{l,j} \) in the current frame by another block of equal size in the reference frame, using motion vector \( m_{l,j} \), can be viewed as quantization of the block \( b_{l,j} \), with which a certain rate and a certain distortion are associated, the problem of rate-constrained motion estimation can be viewed as an optimal bit allocation problem among the blocks of a QT with leaf dependencies or dependent quantizers. Hence, the methods of Section 11.4 (Lagrangian multiplier-based unconstrained optimization using DP within a trellis) are applicable.

The search for the best block match under a motion vector is by far the most computationally expensive part of the optimization. To make the search faster, a slightly suboptimal clustering scheme is used, in which only a subset of possible motion vectors is considered [28].

**Motion Estimation Results**

The QT-based optimal motion estimation scheme compares favorably with TMN4 (an implementation of the H.263 standard), using the same quantizers to encode the DVF. For the QCIF video sequence, the smallest block size of \( 8 \times 8 \) was chosen and the Hilbert scan was modified [28] to cover nonsquare frames. The predicted frame 180 of the Mother and Daughter sequence and its corresponding scanning path are shown in Figures 11.4 and 11.5, respectively.
When compared to the original frame 180, the DFD peak signal-to-noise ratio (PSNR), in the case of optimal QT motion estimation, is 31.31 dB. The corresponding bit rate is 472 bits. Compared to the TMN4 encoder, operating at the same rate and resulting in the PSNR of 30.65 dB, the optimal scheme represents a 0.67-dB improvement in PSNR. The results are more dramatic if, rather than the rate, the PSNR is matched between the two algorithms (at 30.65 dB) (i.e., the dual problem is solved). In this case, the optimal QT-base motion estimator encodes frame 180 with 344 bits, and that is a 26.8% improvement over the 470 bits required by TMN4.
Motion-Compensated Interpolation

The problem of frame interpolation is very important in very-low-bit-rate video coding. The transmission channel capacity, by imposing an upper limit on the bit rate, necessitates a two-pronged approach to video compression: some frames are encoded and transmitted; other frames are not encoded at all, or dropped, and must be interpolated at the decoder. It is common for video codecs to operate at the rate of 7.5 or 10 frames per second (fps) by dropping every third or fourth frame. Without frame interpolation, or with zero-hold frame interpolation, the reconstructed video sequence will appear jerky to a human observer.

The problem of interpolation is ill posed because not enough information is given to establish a metric by which to judge the goodness of a solution. In our context, some frame in the future, \( \hat{f}_N \), and some frame in the past, \( \hat{f}_0 \), of the interpolated frame is all that is available at the decoder. Therefore, since the original skipped frame is not available at the decoder, the MSE minimization-based interpolation is not possible.

What makes this problem solvable is the underlying assumption that changes in the video scene are due to object motion, in particular linear motion. In contrast to the traditional approach, where the motion is estimated for frame \( \hat{f}_N \) with respect to frame \( \hat{f}_0 \) and then projected onto the interpolated frame, here the motion is estimated directly for the interpolated frame. Thus, the problems related to the fact that not all pels of the interpolated frame are associated with a motion vector are avoided. Figure 11.6 demonstrates this idea.

![Diagram of motion-compensated interpolation](image)

**FIGURE 11.6**
Motion-compensated interpolation (of frame 1 from frame 0 and frame 4).

The problem at hand is to find the segmentation of the interpolated frame into blocks \( b_{l,i} \) (QT decomposition) and the associated motion vectors \( m_{l,i} \). The motion, however, is with respect to both reference frames (\( \hat{f}_0 \) and \( \hat{f}_N \)). A block \( b_{l,i} \) in the QT decomposition is interpolated as
follows:

\[
\hat{f}_n(x,y) = \frac{N_2 \hat{f}_0(x - N_1 m_{j,x}, y - N_1 m_{j,y}) + N_1 \hat{f}_N(x + N_2 m_{j,x}, y + N_2 m_{j,y})}{N_1 + N_2}
\]

\forall (x,y) \in b_{i,j}, \tag{11.11}

where 0 \leq n \leq N, m_{j,x} and m_{j,y} are the x and y coordinates of the motion vector \( m_j \) of the block \( b_{i,j} \), and \( N_1, N_2 \) are the temporal distances of the interpolated frame to frames \( f_0 \) and \( f_N \), respectively. This weighted definition leads to a smooth transition of the interpolated frame toward the closer reference frame as the distance between them decreases, causing less jerkiness.

Let \( x_j \) denote the global system state, corresponding to the interpolated block \( b_{i,j} \) undergoing motion \( m_j \). Consistent with the block interpolation formula defined in (11.11), the associated block distortion is defined as follows:

\[
d(x_j) = \sum_{(x,y) \in b_{i,j}} \left( \hat{f}_0(x - N_1 m_{j,x}, y - N_1 m_{j,y}) - \hat{f}_N(x + N_2 m_{j,x}, y + N_2 m_{j,y}) \right)^2,
\]

and the overall distortion is the algebraic sum of block distortions corresponding to the leaves in the chosen QT decomposition. That is,

\[
D(x_0, \ldots, x_{N_{\Theta} - 1}) = \sum_{j=0}^{N_{\Theta} - 1} d(x_k). \tag{11.13}
\]

Clearly, minimizing the frame distortion defined in (11.13) alone over all possible QT segmentations and DVF choices would lead to the frame being segmented into blocks of the smallest size possible. The resulting DVF would be very noisy and have little resemblance to the underlying object motion in a video scene. It is desired for the estimated DVF to possess a measure of smoothness present in the real DVF. It turns out that this goal can be achieved by regularizing the objective function with the total bit rate, that is,

\[
\min_{x_0, \ldots, x_{N_{\Theta} - 1}} \left( D(x_0, \ldots, x_{N_{\Theta} - 1}) + \lambda \cdot R(x_0, \ldots, x_{N_{\Theta} - 1}) \right), \tag{11.14}
\]

where \( \lambda \) is the regularization parameter. Minimizing the above objective function leads to a smooth DVF because, with a differential encoding scheme, there is a strong correlation between the smoothness of the DVF and the bit rate necessary for its encoding. Hence, smoothness is achieved for motion vectors along a scanning path. With the Hilbert scanning path employed, this translates into DVF smoothness in all directions.

The optimal solution to the regularized optimization problem of (11.14) is then found with the help of DP applied to a trellis structure, as explained in Section 11.4.

**Interpolation Results**

The described algorithm has been applied to the compressed Mother and Daughter sequence at the frame rate of 15 fps and a constant frame PSNR of 34.0 dB. Every second frame in this sequence is dropped, thus resulting in a 7.5 fps sequence. Then these dropped frames are reconstructed from the two neighboring frames using the operationally optimal motion-compensated interpolation scheme described in the preceding section.

The issue of selecting the proper regularization parameter \( \lambda \) is addressed in [5]. Here, a \( \lambda \) of 0.01 is used. Figure 11.7 shows the reconstructed frame 86, which was interpolated from frame 84 and frame 88. The resulting DVF and QT segmentation are also overlaid on this figure. The interpolated frame is very similar to the original frame 86 and is only 1 dB lower in PSNR than the reconstructed version, had it been transmitted.
11.5.2 QT-Based Video Encoding

In this subsection the operationally optimal bit allocation scheme among QT segmentation, DVF, and DFD is presented. The overall problem solved here can be stated as that of minimizing the distortion between the original and the reconstructed frames for a given bit budget $R_{\text{max}}$, where motion is estimated with respect to a given reference frame. Hence, the job of the encoder is to optimally allocate the available bit budget to segmentation, motion, and error quantization components.

**Code Structure**

As in the case with optimal motion estimation and interpolation, discussed in the previous subsection, here the QT structure is used for frame segmentation due to its being a compromise between a fixed-block-size approach and a scene-adaptive object-based segmentation. Thus, the original $2^N \times 2^N$ image is decomposed into a hierarchy of square blocks, with the smallest block size $2^{n_0} \times 2^{n_0}$.

Using the same notation as in Section 11.5.1, each square block $b_{l,i}$ is associated with $M_{l,i}$ — the set of all admissible motion vectors, of which $m_{l,i}$ is a member — and $Q_{l,i}$ — the set of all admissible residual error quantizers, of which $q_{l,i}$ is a member. Then a local state $s_{l,i} = [l, i, q_{l,i}, m_{l,i}] \in S_{l,i} = \{l\} \times \{i\} \times M_{l,i} \times Q_{l,i}$ can be defined for block $b_{l,i}$. Consequently, a global state $x$, representing the currently chosen local state, is defined as $x \in X = \bigcup_{l=0}^{n_0} \bigcup_{i=0}^{2^{n_0}-1} S_{l,i}$, where $X$ is the set of all admissible global state values.
For completeness of description, individual block states \( s_{j,1} \) must be enumerated sequentially because the rate function may involve arbitrary-order dependencies resulting from differential encoding of parameters. Hence the code for a predicted and motion-compensated frame consists of a global state sequence \( x_0, \ldots, x_{N_{\Theta L} - 1} \), which represents the left-to-right ordered leaves of a valid QT \( \Theta \).

In this context, the frame distortion is an algebraic sum of block distortions \( d(x_j) \), implemented with the MSE metric, where the blocks correspond to the leaves of the chosen QT decomposition \( \Theta \). That is,

\[
D(x_0, \ldots, x_{N_{\Theta L} - 1}) = \sum_{j=0}^{N_{\Theta L} - 1} d(x_j). \tag{11.15}
\]

Again, as a compromise between complexity and efficiency, a first-order DPCM scheme is used for encoding motion vectors, allowing a first-order dependency between the leaves along the scanning path. The scanning path itself is, for reasons described in Section 11.5.1, the Hilbert curve, recursively generated to fill the frame space [28].

Based on the chosen first-order DPCM along the scanning path, the overall frame rate can be expressed as follows:

\[
R(x_0, \ldots, x_{N_{\Theta L} - 1}) = \sum_{j=0}^{N_{\Theta L} - 1} r(x_{j-1}, x_j), \tag{11.16}
\]

where \( r(x_{j-1}, x_j) \) is the block bit rate, which depends on the encoding of the current and the previous blocks.

In (11.16) we assume that the total frame rate can be distributed among its constituent blocks. This assumption is intuitive in the case of the rate associated with the motion vector component \( r^{DVF}(x_{j-1}, x_j) \) and the residual error quantization component \( r^{DFD}(x_j) \). It turns out that the QT segmentation rate can also be distributed on the block basis. With QT decomposition, only 1 bit is required to signal a splitting decision at each level. Hence, smaller blocks carry the segmentation costs of all of their predecessors. With the sequential scanning order of blocks belonging to the same parent in the QT, the first scanned block is arbitrarily assigned the cost \( r^{SEG}(x_j) \) associated with decomposition up to that level. Thus, the segmentation component of the rate can also be defined on the block basis.

To complete the discussion of the rate, we must also take into account the fact that some blocks in the QT decomposition are not predicted, but rather intra-coded. This may be applicable to newly appearing or uncovered objects that are not found in the reference frame or when the motion model fails to find a good match for a block. When that happens, the intra-coded block’s DC coefficient can be predicted from its predecessor’s along the scanning path. Therefore, the \( r^{DC}(x_{j-1}, x_j) \) component must be added to the total rate. Clearly, it is equal to 0 for predicted blocks, and the task of deciding which blocks are coded intra and which blocks are coded inter is a part of the optimization process at the encoder.

In summary, the block rate, corresponding to the transition from state \( x_{j-1} \) to state \( x_j \) is expressed as follows:

\[
r(x_{j-1}, x_j) = r^{SEG}(x_j) + r^{DFD}(x_j) + r^{DVF}(x_{j-1}, x_j) + r^{DC}(x_{j-1}, x_j). \tag{11.17}
\]

Having defined the distortion and the rate, the problem of joint segmentation, motion estimation, and residual error encoding is posed as a constrained optimization problem as follows:

\[
\min_{x_0, \ldots, x_{N_{\Theta L} - 1}} D(x_0, \ldots, x_{N_{\Theta L} - 1}), \quad \text{subject to: } R(x_0, \ldots, x_{N_{\Theta L} - 1}) \leq R_{\text{max}}. \tag{11.18}
\]
This problem can be viewed as the optimal bit allocation problem among the blocks of a QT with leaf dependencies and, hence, the methods of Section 11.4 apply. In particular, it is converted into the unconstrained minimization problem using the Lagrangian multiplier method,

\[ J_\lambda (x_0, \ldots, x_{N_\Theta-1}) = D(x_0, \ldots, x_{N_\Theta-1}) + \lambda \cdot R(x_0, \ldots, x_{N_\Theta-1}), \]  

(11.19)

and dynamic programming, also discussed in Section 11.4, is used to find the optimal solution. The resulting optimal solution is a sequence of states \([x_0^*, \ldots, x_{N_\Theta-1}^*]\).

Graphically, the DP algorithm is illustrated by Figure 11.8. In it, each node (black circle) represents a particular state \(x_j\) the corresponding block \(b_{l,i}\) is in. The lines connecting these states correspond to the possible scanning orders of a Hilbert curve, and weights \(j_\lambda(x_{j-1}, x_j) = d(x_j) + \lambda \cdot r(x_{j-1}, x_j)\) are associated with each transition. Then the problem of optimal resource allocation among segmentation, motion, and DFD quantization can be stated as that of finding the shortest path in the trellis from \(S\) to \(T\), the two auxiliary states. In the implementation, the maximum block size was \(32 \times 32\) and the minimum block size was \(8 \times 8\), with a consequence that no segmentation information for blocks larger than \(32\) and smaller than \(8\) needed to be sent to the decoder. Other implementation details can be found in [28]–[30].

**Results**

The encoder presented here is compared with TMN4, which is an H.263 standard implementation. Both algorithms were tested on 200 frames of the Mother and Daughter sequence, down-sampled by a factor of 4 in the time axis, with the quantizer step size QP set to 10 in TMN4. In the first test, the \(\lambda\) parameter of the optimal algorithm was adjusted for each frame so that the resulting distortion matched that produced by TMN4 at every encoded frame. The resulting curves are shown in Figure 11.9. In the second test, the \(\lambda\) parameter of the optimal algorithm was adjusted for each frame so that the resulting rate matched that produced by TMN4 at every encoded frame. The resulting curves are shown in Figure 11.10. Clearly, the optimal approach significantly outperforms H.263 in both experiments. In the matched distortion case, the average frame bit rate was reduced by about 25%, and, in the matched rate case, the average PSNR distortion was increased by about 0.72 dB.
11.5.3 Hybrid Fractal/DCT Image Compression

This section describes the application of the rate-distortion techniques to the hybrid fractal/DCT image compression. Drawing on the ability of DCT to remove interpixel redundancies and on the ability of fractal transforms to capitalize on long-range correlations in the image, the hybrid coder performs an optimal, in the rate-distortion sense, bit allocation among coding parameters. An orthogonal basis framework is used within which an image segmentation and a hybrid block-based transform are selected jointly.

Problem Formulation

Within the chosen fractal/DCT framework, the problem to be solved is that of simultaneous segmentation of an input image into blocks of variable sizes, and, for each, to find a code
in such a way that any other choice of segmentation and coding parameters would result in
greater distortion for the same rate, or vice versa. Problems of this type are discussed in
Section 11.2 and the corresponding solution tools in Section 11.4. In this context, for a given
image $x$, we want to solve the following optimization problem:

$$\min_{s \in \mathcal{S}, c \in \mathcal{C}_s} D(x_{s,c}, x) \quad \text{subject to:} \quad R(x_{s,c}) \leq R_{\text{max}},$$

(11.20)

where $x_{s,c}$ is the encoded image; $D$ the distortion metric; $s$ a member of the set of all possible
image segmentations $\mathcal{S}$; $c$ a member of $\mathcal{C}_s$, the set of all possible codes given segmentation $s$; $R$ the bit rate associated with segmentation $s$ and code $c$; and $R_{\text{max}}$ the target bit budget. The
distortion metric chosen here is the MSE.

**Fractal Basics**

Fractal image coding takes advantage of image self-similarities on different scales. That
is, instead of sending quantization indices of transformed image subblocks, a fractal coder
describes the image as a collection of nonexpansive transformations onto itself. Most fractal
algorithms, beginning with Jacquin’s implementation [8], break an image into nonoverlapping
square regions, called ranges. Each range block $r_i$ is encoded by a nonexpansive transformation
$T_i^*$ that operates on the entire original image $x$ and maps a domain block $d_i$, twice the size of
the range block and located elsewhere in the image, onto $r_i$. The job of the encoder is to find
a transformation $T_i$ that minimizes the collage error. That is,

$$T_i^* = \arg \min_{T_i \in \Theta} \| r_i - T_i(x) \|,$$

(11.21)

where $\Theta$ is the pool of available transforms. The whole transformation $T$ is a sum of partitioned
transformations,

$$T(x) = \sum_{i=1}^{N} T_i^*(x), \quad x = \sum_{i=1}^{N} r_i,$$

(11.22)

where $N$ is the number of partitions or ranges.

The Collage Theorem establishes an upper bound on the reconstruction error of the decoded
image as a function of the collage error and $s$, the contractivity of $T$. Specifically,

$$d(x, x_T) \leq \frac{1}{1 - s} \cdot d(x, T x),$$

(11.23)

where $x_T$ is the decoded image under transformation $T$.

Transformations $T_i$ are restricted to a set of discrete contractive affine transformations operating on $x$. Following the notation used in [21], each $T_i$ has the following structure:

$$T_i(x) = \beta_i P_i D_i I_i F e t_i x + t_i,$$

(11.24)

where $F e t_i$ is a transformation matrix that fetches the correct domain block, $I_i$ applies one of
the standard eight isometries, $D_i$ is the decimation operator that shrinks the domain block
to the range block size, $P_i$ is the place operator that places the result in the correct region
occupied by the range block, $\beta_i$ is a scalar, and $t_i$ is a constant intensity block. Hence, in
analogy with vector quantization (VQ), various permutations of $F e t_i$, $I_i$, and $\beta_i$ represent
the codebook. Implementation details on the specific choice of the variable-length coding (VLC)
for the various parameters of (11.24) can be found in [13].

Blocks for which a good approximation, under a contractive transformation, can be found
elsewhere in the image, can be efficiently encoded using the fractal transform. The self-
similarity assumption, which is central to fractals, however, may not be justified for all blocks.
In this case, spending more bits on the fractal transform by employing more isometries or finer quantizers is not efficient [4, 25].

The discrete cosine transform (DCT) has been the transform of choice for most codecs due to its decorrelation and energy compaction properties. Complicated image features, however, require a significant number of DCT coefficients to achieve good fidelity. The coarse quantization of these coefficients results in blocking artifacts and unsharp edges.

The coder described here is a hybrid in that it not only adaptively selects which transform (fractal or DCT) to use on any given block, but also can use them jointly and in any proportion. Thus it capitalizes on the ability of the fractal transform to decorrelate images on the block level and on the ability of the DCT to decorrelate pixels within each block. The optimization techniques of Section 11.4 are applied within the chosen framework, resulting in a code that is optimal in the operational sense.

The decoder reconstructs an approximation to the original image by iterative application of the transformation defined by (11.22) to any arbitrary image \( x_0 \). Although the Collage Theorem, expressed by (11.23), guarantees the eventual convergence of this process to an approximation with a bounded error, the number of these iterations may be quite large.

**Segmentation**

The goal of image segmentation in the context of compression is to adapt to local characteristics of the image. Segmenting the image into very small square blocks or into objects of nonsquare shapes, while accomplishing this goal, is also associated with a high cost of description. Here, the set of all possible segmentations \( \mathcal{S} \) is restricted to be on the QT lattice as a compromise between local adaptivity and simplicity of description. For a 256 × 256 input image it is a three-level QT with a maximum block size of 16 × 16 pixels and a minimum size of 4 × 4 pixels. At each level of the QT only 1 bit is required to signal a splitting decision, with no such bit required at the lowest level.

**Code Structure**

As mentioned in Section 11.5.3, the transform employed for encoding range blocks consists of the fractal and DCT components. The hybrid approach allows for more flexibility at the encoder, and the various components of the overall transform are designed to complement each other. This idea leads naturally to the concept of orthogonality.

The presence of the DCT component in the overall transform, as well as the fact that the frequency domain interpretation lends itself naturally to the concept of orthogonality and energy compactness, makes it more convenient to perform the collage error minimization of (11.21) in the DCT domain. This is done by applying the DCT to both the range block \( r_i \) under consideration and all candidate-decimated domain blocks.

It is convenient to cast the problem of finding the overall transform as that of vector space representation. Vectors are formed by zigzag-scanning square blocks, as shown in Figure 11.11. Hence, minimizing the collage error with respect to a range vector \( \vec{r}_i \) is equivalent to finding its best approximation in a subspace spanned by a combination of transformed domain vectors and some fixed (image-independent) vectors \( \vec{f}_k \).

In agreement with the terminology used in [23], let the vector \( \vec{r}_i \), of size \( M^2 \times 1 \), represent the DCT coefficients of range block \( i \), of size \( M \times M \), scanned in zigzag order. Similarly, the vector \( \vec{d}_i \) comes from the chosen domain block, after decimation, DCT, and the application of isometry operators. The fractal component of the overall transform can then be expressed as follows:

\[
\vec{d}_i = P_i \ Z_i \ DCT_i \ D_i \ I_i \ Feti \ x \ ,
\]  

(11.25)
where \( P_i, D_i, I_i, \) and \( Feti \) are defined as in (11.24) and \( DCT_i \) and \( Z_i \) are the block DCT and zigzag-scan operators, respectively. Let the intensity translation term be represented by a linear combination of \( N_i - 1 \) fixed vectors \( \tilde{f}_{ik} \), of size \( M^2 \times 1 \), where the subscript \( i \) indicates that the pool of available fixed vectors and the total number of them may be locally adaptive to the range vector. Mathematically, the range vector \( \tilde{r}_i \) is then approximated by \( N_i \) vectors as follows:

\[
\tilde{r}_i \approx \beta_i \cdot \tilde{d}_{io} + \sum_{k=0}^{N_i-2} c_{ik} \cdot \tilde{f}_{ik},
\]

(11.26)

where \( \tilde{d}_{io} \) is the projection of \( \tilde{d}_i \) onto the orthogonal complement of the subspace spanned by vectors \( \tilde{f}_{ik} \) for \( k = 0, \ldots, N_i - 2 \), which are themselves orthogonal to each other. Making components of the overall transform orthogonal to each other carries many benefits [16, 22]. These include fast convergence at the decoder, noniterative determination of scaling and intensity translation parameters, no restriction on the magnitude of the scaling coefficient, and continuity of the magnitude of the translation term between neighboring blocks.

In this implementation a bank of fixed subspaces is used to model a range vector of a given size. To illustrate how a fixed subspace is formed from the coefficients of a block DCT, let us, for simplicity, consider a \( 2 \times 2 \) block of DCT coefficients. A subspace of dimension 3 (with the full space of dimension 4) is then formed from the low-frequency coefficients as shown in Figure 11.12. Each \( \tilde{f}_{ik} \) corresponds to one coefficient in the two-dimensional DCT of size \( M \times M \). Each vector \( f_{ik} \) has zeros in all positions, except the one corresponding to the

FIGURE 11.11
Zigzag scan of a square block.

FIGURE 11.12
Mapping of selected DCT coefficients into basis vectors.
order in which the DCT coefficient it represents was scanned, where it has 1. Since in larger blocks more DCT coefficients tend to be significant, the fixed space used for the encoding of a 16 × 16 range block is allowed to be of a higher dimension than that of a 4 × 4 block. With the limited number of available subspaces for each block size, only the subspace index, and not the positions of individual nonzero coefficients, needs to be sent to the decoder. Figure 11.13 shows the subspaces allowed for encoding range blocks of size 4 × 4 (range vectors of size 16 × 1). The banks of subspaces for range blocks of sizes 8 × 8 and 16 × 16 are defined similarly and the details on this, as well as on the VLCs used, can be found in [16].

The job of the encoder is then, for each range vector \( \bar{r}_i \), to select the fixed space dimension \( N_i - 1 \), the set of coefficients \( c_{ik} \), the domain block to be fetched by \( F_{eti} \), the isometry \( I_i \), and the scaling coefficient \( \beta_i \). As a result of orthogonalization, the fixed subspace coefficients \( c_{ik} \) will carry low-frequency information and fractal component parameters will carry high-frequency information.

**Directed Acyclic Graph (DAG) Solution**

The DC coefficients of neighboring blocks in an image decomposition exhibit high correlation. In this formulation, coefficient \( c_{i0} \) corresponds to the quantized DC value of the range block in question. Hence, differential encoding must be introduced between adjacent blocks to take advantage of this redundancy. The Hilbert curve is known to satisfy certain adjacency requirements [28] and is efficient for predictive coding. For a 256 × 256 input image, a sixth-order Hilbert curve is used.

If we let each node represent a block in the QT decomposition of the image, and define a transition cost \( g_{il,j} \) as the cost of encoding range block \( r_i \) with range block \( r_j \) as its predecessor, then the overall problem of finding the optimal segmentation and the hybrid fractal/DCT code can be posed as that of finding the shortest path through the leaves of the QT decomposition or trellis, with each leaf having one to three possible codes, corresponding to one to three possible predecessors of a block in our Hilbert curve.

Clearly, the optimal scanning path possesses the optimal substructure property (i.e., it consists of optimal segments). This motivates the use of dynamic programming in the solution. Refer to [28] and Section 11.4 for more details on the use of DP in problems of this type.

**Fractal Compression Results**

Performance of the operationally optimal hybrid fractal/DCT algorithm is compared to JPEG, which is one of the most popular DCT-based compression schemes. Figure 11.14 shows the ORD curve obtained with this approach when compressing a 256 × 256 Lena image. JPEG’s ORD curve is also shown on this plot. An improvement of 1.5 to 3.0 dB is achieved across the range of bit rates.

Figures 11.15 and 11.16 demonstrate the improvement in quality, over JPEG, for the same bit rate (0.20 bpp). Figure 11.17 shows the optimal segmentation as determined by this encoder. Efficiency is achieved by using larger block sizes in relatively uniform areas and smaller block sizes in edgy areas. Overall, the fractal component of the transform, representing high-frequency information, used about 30% of the available bit budget. This, coupled with
the rate-distortion optimized scanning path, segmentation, and code selection, resulted in significant gains in the quality of the reconstructed image.

11.5.4 Shape Coding

In this section we show how rate-distortion operationally optimal techniques can be applied to the problem of object shape coding. Interest in this problem is motivated by a growing explosion in new multimedia applications, including, but not limited to, video conferencing,
interactive multimedia databases, film authoring, etc. They all have in common the requirement that video information be accessible on an object-by-object basis.

Commercial video compression standards, such as MPEG-1, MPEG-2, H.261, and H.263, are block-based codecs. They segment a video scene into fixed blocks of predetermined sizes and achieve compression by quantizing texture and motion vectors. This format of representation is not natural for the above-mentioned applications, since there is no clear separation of one object from another and from the background in a generated bitstream. It leads to an unnecessary waste of bits describing a background that carries no information. But it is also inefficient in terms of accessing encoded information. For example, in pattern recognition applications objects are detected through their boundaries, which are not explicitly available in block-based bitstreams.

The emerging MPEG-4 and MPEG-7 multimedia coding standards are designed to address these new requirements. Although it is not clear whether object-based treatment of a video scene is justified in terms of coding efficiency, it is in some cases a requirement from the application point of view. In an object-oriented coder, bits must be efficiently allocated among bitstream components (segmentation, motion, shape, texture) and then within each component. Although ORD optimal joint resource allocation among and within these components in an object-oriented coder remains an elusive goal, here we address the shape coding aspect of this problem.

Shape information plays the central role in object description. Efforts at its efficient representation, which intensified as a result of the MPEG-4 standardization, can be classified into two categories [9]. The first consists of bitmap-based coders, which can be further broken down into context-based [1] and modified read fax-like [36] coders. The baseline-based shape coder [12] and vertex-based polynomial coder [7, 19] belong to the second category. However, arguably, the bitmap-based coders defeat the goal of object orientation, since in them the shape information is not explicit.

In what follows we describe an intra-mode vertex-based boundary encoding scheme and
FIGURE 11.17
Optimal segmentation ($R = 0.44$ bpp, PSNR = 30.33 dB).

how it is optimized using the techniques of Section 11.4. Implementation details can be found in [33]. An inter-mode vertex-based boundary encoding scheme is derived in [15].

Algorithm

The original boundary is approximated by second-order connected spline segments. A spline segment is completely defined by three consecutive control points ($p_{u-1}$, $p_u$, $p_{u+1}$). It is a parametric curve (parameterized by $t$), which starts at the midpoint between $p_{u-1}$ and $p_u$ and ends at the midpoint between $p_u$ and $p_{u+1}$, as $t$ sweeps from 0 to 1. Mathematically, the second-order spline segment used is defined as follows:

$$
\hat{Q}_u (p_{u-1}, p_u, p_{u+1}, t) = \begin{bmatrix} t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 0.5 & -1.0 & 0.5 \\ -1.0 & 1.0 & 0.0 \\ 0.5 & 0.5 & 0.0 \end{bmatrix} \begin{bmatrix} p_{u-1},x & p_{u-1},y \\ p_{u},x & p_{u},y \\ p_{u+1},x & p_{u+1},y \end{bmatrix},
$$

(11.27)

where $p_{i,x}$ and $p_{i,y}$ are, respectively, the vertical and horizontal components of point $p_i$.

Besides solving the interpolation problem at the midpoints, the definition of the spline used makes it continuously differentiable everywhere, including the junction points. Segment continuity is ensured because the next spline segment, ($p_u$, $p_{u+1}$, $p_{u+2}$), will originate at the end of the current segment. Placing the control points appropriately, a great variety of shapes can be approximated, including straight lines and curves, a property that makes splines a very attractive building block for the contour approximation problem. The operationally optimal straight line shape approximation was derived in [31].

In order to fit a continuous spline segment to the support grid of the original boundary, spline points are quantized toward the nearest integer value. Thus the solution to our shape approximation problem is an ordered set of control points, which, based on a particular definition of the distortion and rate (discussed below), and for a given rate-distortion trade-off $\lambda$, results in a rate-distortion pair ($R$, $D$).
Theoretically, the ordered set of control points can be composed of points located anywhere in the image. Most of them, however, are highly unlikely to belong to the solution, as they are too far from the original boundary, and distortions of more than several pixels are not tolerable in most applications. For this reason, and also to decrease computational complexity, we exclude those points from consideration. What remains is the region of space, shown in Figure 11.18, centered around the original boundary and termed the admissible control point band, to which candidate control points must belong. Pixels in this band are labeled by the index of the closest original boundary pixel (boundary pixels themselves are ordered and labeled). Consecutive control points must be of the increasing index, thus ensuring that the approximating curve can only go forward along the original boundary.

Distortion

Measuring the distortion between an original and approximating boundary is a nontrivial problem. In the minimum–maximum distortion problem [31], stated in Section 11.3, the metric is a binary function, evaluating to zero in case the approximation is inside the distortion band of width $D_{\text{max}}$, and evaluating to infinity in case some portion of the approximating curve lies outside this band. This metric may be useful when fidelity of approximation must be guaranteed for all pixels.

The minimum total distortion problem, solved optimally in [14, 17], is based on a global distortion measure, where local errors are not explicitly constrained. This measure was used in the MPEG-4 standardization process to compare performances of competing algorithms and is also used here explicitly in the optimization process. It is defined as follows:

$$D = \frac{\text{number of pixels in error}}{\text{number of interior pixels}}$$ (11.28)

FIGURE 11.18
Admissible control point band.

© 2001 CRC Press LLC
A pixel is judged to be in error if it is in the interior of the original boundary but not in the interior of the approximating boundary, or vice versa. It should be noted that the application of this distortion metric in the optimization process is a stark departure from techniques proposed previously in the literature, in which ad hoc algorithms were simply evaluated after their execution, with equation (11.28). A variation of this definition was used in [14], where in the numerator the area between the original boundary and its continuous approximation was used.

Regardless of the distortion criterion, in order to define the total boundary distortion, a segment distortion needs to be defined first. For that, the correspondence between a segment of the approximating curve and a segment of the original boundary must be established. Figure 11.19 illustrates this concept. In it, the midpoints of the line segments \((p_{u-1}, p_u)\) and \((p_u, p_{u+1})\),

![FIGURE 11.19](image)

**Area between the original boundary segment and its spline approximation (circles).**

\(l\) and \(m\), respectively, are associated with the points of the boundary closest to them, \(l'\) and \(m'\). When more than one boundary pixel is a candidate, we select the one with the larger index. This ensures that the starting boundary pixel of the next segment coincides with the last boundary pixel of the current segment. That is, the segment of the original boundary \((l', m')\) is approximated by the spline segment \((l, m)\). In order to ensure that some pixels on the edge between two adjacent distortion areas are not counted twice, we exclude points on the line \((m, m')\) from being counted among the distortion pixels (shown in circles).

Let us now define by \(d(p_{u-1}, p_u, p_{u+1})\) the segment distortion, as shown in Figure 11.19. Based on the segment distortions, the total boundary distortion is therefore defined by

\[
D(p_0, \ldots, p_{N_p-1}) = \sum_{u=0}^{N_p} d(p_{u-1}, p_u, p_{u+1}),
\]

(11.29)

where \(p_{-1} = p_{N_p+1} = p_{N_p} = p_0\) and \(N_p\) is the number of control points.

**Rate**

Consecutive control point locations along an approximating curve are decorrelated using a second-order prediction model [9]. Each control point is described in terms of the relative angle \(\alpha\) it forms with respect to the line connecting two previously encoded control points, and by the run length \(\beta\) (in pixels), as shown in Figure 11.20A. The range of values of the angle \(\alpha\) is taken from the set \([-90^\circ, -45^\circ, 45^\circ, 90^\circ]\), thus requiring only 2 bits (Figure 11.20B). The rationale for excluding the angle of \(0^\circ\) is that that orientation is unlikely, since it can be
achieved by properly placing the preceding control point, thus with fewer bits. The exception to this scheme is the encoding of the first and second control points, for which predictive coding does not exist. They are encoded in an absolute fashion and with a 3-bit angle, respectively. To ensure that a closed contour is approximated by a closed contour, we force the control point band of the last boundary pixel to collapse to just the boundary pixel itself, thereby making the approximating curve pass through it. It should be noted, however, that, in general, this way of encoding consecutive control points \((\text{run, angle})\) is somewhat arbitrary and other predictive schemes or a different range of \(\alpha\) could have been used without loss of generality.

If \(r(p_{u-1}, p_u, p_{u+1})\) denotes the segment rate for representing \(p_{u+1}\) given control points \(p_{u-1}, p_u\), then the total rate is given by

\[
R(p_0,\ldots,p_{N_p-1}) = \sum_{u=0}^{N_p-1} r(p_{u-1}, p_u, p_{u+1}) .
\]

Here we do not restrict the possible locations of \(p_{u+1}\), given \(p_{u-1}\) and \(p_u\), and let the encoder determine the locally most efficient VLC for the vector \(p_{u+1} - p_u\).

**DAG Solution**

Having defined the distortion and the rate, we now solve the following rate-constrained optimization problem:

\[
\min_{p_0,\ldots,p_{N_p-1}} \quad D(p_0,\ldots,p_{N_p-1}), \quad \text{subject to:} \quad R(p_0,\ldots,p_{N_p-1}) \leq R_{\text{max}} ,
\]

where both the location of the control points \(p_i\) and their overall number \(N_p\) have to be determined.

Let us define an incremental cost of encoding one spline segment as

\[
w(p_{u-1}, p_u, p_{u+1}) = d(p_{u-1}, p_u, p_{u+1}) + \lambda \cdot r(p_{u-1}, p_u, p_{u+1}) .
\]
The overall Lagrangian cost function can then be written as

\[ J_{\lambda} (p_0, \ldots, p_{N-1}) = \sum_{u=1}^{N-1} w(p_{u-1}, p_u, p_{u+1}) + w(p_{N-1}, p_N, p_{N+1}) = J_{\lambda} (p_0, \ldots, p_{N-2}) + w(p_{N-1}, p_N, p_{N+1}). \] (11.33)

This problem now can be cast as the shortest path problem in a graph with each consecutive pair of control points playing the role of a vertex and incremental costs \( w() \) serving as the corresponding weights [9]. The problem is then efficiently solved using the techniques of Section 11.4.

**VLC Optimization**

The operationally optimal shape-coding algorithm described here can claim optimality only with respect to the chosen representation of control points of the curve. That is, our solution is operationally optimal when the encoding structure \( (\text{run, angle}) \) and its associated VLC are fixed, which is equivalent to solving the following optimization problem:

\[ \{p_0^*, \ldots, p_{N-1}^*\} = \arg \min_{p_0, \ldots, p_{N-1}} J_{\lambda}^* (p_0, \ldots, p_{N-1}) \mid \text{VLC}. \] (11.34)

Here, we take the operationally optimal approach one step further, and remove the dependency of the ORD on an ad hoc VLC used. Our goal is to compress the source whose alphabet consists of tuples \( (\text{run, angle}) \), exhibiting first-order dependency, close to its entropy. Therefore, the problem can be stated as follows:

\[ \{p_0^*, \ldots, p_{N-1}^*\} = \arg \min_{p_0, \ldots, p_{N-1}} \min_{f \in F} J_{\lambda}^* (p_0, \ldots, p_{N-1}) \text{ \mid VLC}, \] (11.35)

where the code is operationally optimal over all \( f \) belonging to the family of the probability mass functions, \( F \), associated with the code symbols. Hence, two problems need to be solved jointly: the distribution model, \( f \), and the boundary approximation based on that model. Clearly, as \( f \) changes in the process of finding the underlying probability model, symbols generated by the operationally optimal coder are also changed. As is typically done with such codependent problems, the two solutions are arrived at in an iterative fashion [27]. The overall iterative procedure is shown in Figure 11.21. Iterations begin with the optimal boundary encoding algorithm, described in the preceding sections, compressing the input boundaries based on some initial conditional distribution of the \( (\text{run}_i, \text{angle}_i) \) symbol, conditioned on the previously encoded \( \text{run}_{i-1} \).

Having encoded the input sequence at iteration \( k \), based on the probability mass function \( f^k() \), we use the frequency of the output symbols to compute \( f^{k+1}() \), and so on. It is straightforward to show that the total Lagrangian cost is a nonincreasing function of the iteration \( k \). Thus each iteration brings \( f \) closer to the local minimum of \( J_{\lambda}(\cdot) \), and \( f^k \) converges to \( f^M \), where \( M \) is the number of the last iteration. The iterations stop when \( |J_{\lambda}^k(\cdot) - J_{\lambda}^{k+1}(\cdot)| \leq \epsilon \).

The local minimum in this context should be understood in the sense that a small perturbation of the probability mass function \( f \) will result in increases in the cost function \( J_{\lambda}(\cdot) \).

**Shape-Encoding Results**

Figure 11.22 shows the ORD curve resulting from the application of the described iterative algorithm to the SIF sequence, Kids. As shown in Figure 11.21, after convergence the symbols were arithmetically encoded. For comparison purposes, ORD curves with no VLC optimization and the rate-distortion performance of the baseline method, which is the most
efficient method among competing algorithms in MPEG-4, are also shown. The distortion axis represents the average of the \( D \)’s defined in (11.28) for one frame, over 100 frames. The approach described here is by far superior to both the contour-based and pixel-based algorithms in [9]. It also outperforms the fixed-VLC area-based approach [14] (shown with squares) and the fixed-VLC pixel-based encoding (shown as the VLC1 and VLC3 curves).

**FIGURE 11.22**
Rate-distortion curves.

*Figure 11.23* shows a sample frame in the sequence and the error associated with the optimal solution. If the uncompressed boundary requires 3 bits per boundary pixel, the approximation
shown is a 7.1:1 compression. It can be seen that the optimal way to encode some small objects is not to encode them at all, as demonstrated by the space between the legs of the kid on the left, shown in white as an erroneous area. Algorithms not optimized in the rate-distortion sense lack the ability to discard small or noise-level objects, which is required when operating at very low bit rates.

11.6 Conclusions

In this chapter we investigated the application of rate-distortion techniques to image and video processing problems. The ORD theory was presented along with several useful mathematical tools commonly used to solve discrete optimization problems. We concluded by providing several examples from different areas of multimedia research in which successful application of RD techniques led to significant gains in performance.

References


