ERROR CONCEALMENT IN ENCODED IMAGES AND VIDEO

A Thesis
Submitted to the Faculty

of

Purdue University
School of Electrical Engineering

by

Paul Salama

In Partial Fulfillment of the
Requirements for the Degree

of

Doctor of Philosophy

August 1999

To My Parents

ACKNOWLEDGMENTS

I wish to express my sincere gratitude to Professor Edward J. Delp for his guidance and support. I have truly benefited from working with him, and I greatly appreciate his care and dedication in constructively criticizing my work and my thesis.

I would also like to express my thanks to Professor Ness B. Shroff for his guidance. I have truly benefited from his co-supervision of my work.

I wish to also thank the other members of my committee, Prof. Edward J. Coyle and Prof. Bradley Lucier for their support.

I would also like to express my deepest thanks to Dr. Christine Podilchuck and Lucent Technologies for partially supporting this work.

Finally, I would like to thank my family for their continuous and unwavering support.

DISCARD THIS PAGE

TABLE OF CONTENTS

Page

Appendix

Page

DISCARD THIS PAGE

LIST OF TABLES

DISCARD THIS PAGE

LIST OF FIGURES

Appendix
Figure Page

Appendix
Figure Page

Appendix
Figure Page

Appendix
Figure Page

Appendix
Figure Page

Appendix
Figure Page

Appendix
Figure Page

Appendix
Figure Page

Appendix
Figure Page

Appendix
Figure Page

Appendix
Figure Page

Appendix
Figure Page

Appendix
Figure Page

Appendix
Figure Page

ABSTRACT

Salama, Paul, Ph.D., Purdue University, August 1999. Error Concealment in Encoded Images and Video. Major Professors: Edward J. Delp, and Ness B. Shroff.

When transmitting compressed video over a data network, one has to deal with how channel errors affect the decoding process. This is particularly problematic with data loss or erasures. In this thesis we describe techniques to address this problem in the context of networks where channel errors or congestion can result in the loss of entire macroblocks when MPEG video is transmitted.

We propose a technique for packing compressed data into packets, with the aim of detecting the location of missing macroblocks in the encoded video stream. This technique also permits proper decoding of correctly received macroblocks, and thus prevents the loss of packets from affecting the decoding process.

We then describe spatial and temporal techniques for the recovery of lost macroblocks. Spatial restoration is performed by modeling the image as a Markov Random Field (MRF) and then obtaining the maximum a posteriori estimate of the missing data. We also describe a technique that can be implemented in real-time.

In temporal reconstruction, we classify the available motion vectors into 9 classes via a ternary tree. Each class is assigned a cost and the vectors belonging to the class with minimum cost are modeled as an MRF. The map estimate of the motion vector belonging to the class having the smallest cost is obtained. If there is more than one map estimate for a missing motion vector then the vector that "best" preserves macroblock boundaries is chosen.

We also propose the use of unequal error protection for protecting images coded by means of rate scalable algorithms, such as the Embedded Zerotree Wavelet algorithm. More important data such as IZ (isolated zero) coefficients and ZTR (zerotree) coefficients are provided with stronger protection as compared to the NEG (negative significant) and POS (positive significant) coefficients. The coded stream are then interleaved prior to transmission.

1. INTRODUCTION

It is envisioned that one of the most important network applications will involve transmitting digital video [1]. During periods of network congestion packets may be dropped, badly degrading the quality of the video as a result of the missing data. Since retransmission is not a viable option for real-time multimedia applications, error concealment algorithms need to be developed. These algorithms estimate the missing data from the received video in an effort to conceal the effect of channel impairements. Furthermore, these algorithms will have to be integrated into the decoder hardware, and hence be simple enough to be implemented in real time.

In this thesis we consider two major issues for error concealment of MPEG video sent over data networks: how to efficiently pack MPEG video traffic, and how to restore or conceal image data lost due to packet loss. We also explore the use of unequal error protection for protecting still images coded by means of rate scalable algorithms, such as the Embedded Zerotree Wavelet algorithm, prior to transmitting them across a data network.

It is necessary to develop a scheme for packing MPEG Transport streams into network packets so that the decoder can detect where in the data stream the loss of information has occurred. This is important not only for locating the errors in order to reconstruct the lost data, but also for ensuring that the decoder does not improperly decode the undamaged portion of the sequence. For example, if the decoder is oblivious to packet losses, then it may lose synchronization and hence be unable to properly display several subsequent frames. Finally, the packing scheme must incur a very low overhead increase in data rate.

Once we are able to identify the portions of the sequence that are affected by packet loss, appropriate post processing techniques need to be developed to reconstruct the video sequence. Currently the approaches utilized for signal restoration/error concealment are either active concealment or passive concealment. In active concealment, error control coding techniques are used along with retransmission. Since extra data must be transmitted, it is sometimes necessary to reduce the source coder's data rate to avoid increasing network congestion. Active concealment has the advantage of permitting perfect reconstruction at the decoding end, if the amount of data lost is not significant, i.e. within the parameters of the error control coding scheme. In addition, unequal error protection can be provided by varying the number of bits used according to the priority of the data being protected [2, 3]. In passive concealment the video stream is post-processed to reconstruct the missing data. Although passive concealment does not result in perfect reconstruction of the lost data, it is necessary in many applications where error control coding cannot be used because of a high level of overhead, problem with compliance with video transmission standards, or when active concealment itself fails. All video decoders (e.g. a set-top box) will have to implement, in real-time, some form of concealment. The error concealment techniques discussed here are passive techniques used for MPEG compressed video. A general overview with an extensive bibliography of the various error concealment methods can be found in [4].

The error concealment algorithms discussed in this thesis are categorized as being either spatial or temporal in nature. The spatial techniques rely on pixel data within a current damaged frame and a Markov Random Field (MRF) model [5, 6, 7] of the frame to restore any damaged areas of the frame. Similarly, estimates for missing motion vectors are obtained by modeling the motion field as a MRF and finding the maximum a posteriori (MAP) estimate of each missing motion vector given its neighboring motion vectors. We also show that the widely used huerisitic technique based on averaging the motion vectors of neighboring macroblocks [8] is a special case of our MAP estimate. We also describe a temporal-spatial method for restoring

damaged macroblocks based on the use of a ternary tree for classifying the available motion vectors, and the MRF model for the corrupted frame.

We also investigate the use of unequal error protection [3] to provide different protection levels for the coefficients of coded still images that have been encoded by means of the Embedded Zerotree Wavelet algorithm [9]. It is observed that the ZTR (zerotree) and IZ (isolated zero) coefficients are more important than the POS (positive significant) and NEG (negative significant) coefficients, and will impact the quality of the decoded image more. Thus, two levels of protection can be applied to both groups of coefficients.

In Chapters 2 and 3 we provide a brief overview of the ATM protocol and the MPEG video compression standards, respectively. In addition, an overview of MPEG Transport streams is provided in Chapter 3.

Previous work addressing the packing of network packets with video data as well as error concealment algorithms is presented in Chapter 4. In addition, error resilience measures used in the standards are described in Chapter 5. In Chapter 6, we discuss the problems that arise when packing MPEG-1 video into data packets, and propose a new packing scheme.

We address the concealment of lost macroblocks resulting from packet loss in Chapter 7. Several approaches to lost signal restoration in encoded MPEG video sequences are described. These include modeling the original image (decompressed) as a Markov Random Field (MRF), and using the proposed model to spatially interpolate missing macroblocks via MAP estimation. Our temporal-spatial approach for estimating missing motion vectors is also discussed in Chapter 7.

Finally, the use of unequal error protection for protecting the coefficients of still images coded by means of Embedded Zerotree Wavelet algorithm is given in Chapter 8.

In all of our experiments we assumed that the protocol for transmitting video over networks is the Asynchronous Transfer Mode (ATM) protocol.

2. ASYNCHRONOUS TRANSFER MODE (ATM) OVERVIEW

Asynchronous transfer mode (ATM) [10, 11, 12] has been chosen as the target protocol for broadband integrated services digital network (B-ISDN) because it offers a flexible transfer capability for a wide variety of data. These include video sequences, still images, and audio . In an ATM network, data is buffered as it arrives, segmented into blocks of fixed length, and inserted into what is known as an ATM cell. Cells are then transmitted across the network to the intended destination. Cells from different sources are multiplexed onto the channel thereby increasing the efficiency of the network. A cell, as shown in Figure 2.1, is 53 bytes in length and consists of an information field carrying user information (cell payload), and a header containing information used for routing and error detection purposes [10]. ATM networks can operate in either of a connection oriented mode, in which cells are transmitted over designated paths, or a connection less mode [10].

2.1 ATM Layer

This layer is concerned with transporting data across a network. Its responsibilities include cell multiplexing and demultiplexing, virtual path identifier (VPI) and virtual channel identifier (VCI) translation, cell header generation and extraction, and generic flow control.

To transport data, ATM uses virtual connections, where each connection consists of a virtual path (VP), having a virtual path identifier (VPI), and a virtual channel (VC) having a virtual channel identifier (VCI). In the transmit direction, the cell multiplexing function combines cells from individual virtual paths and virtual channels

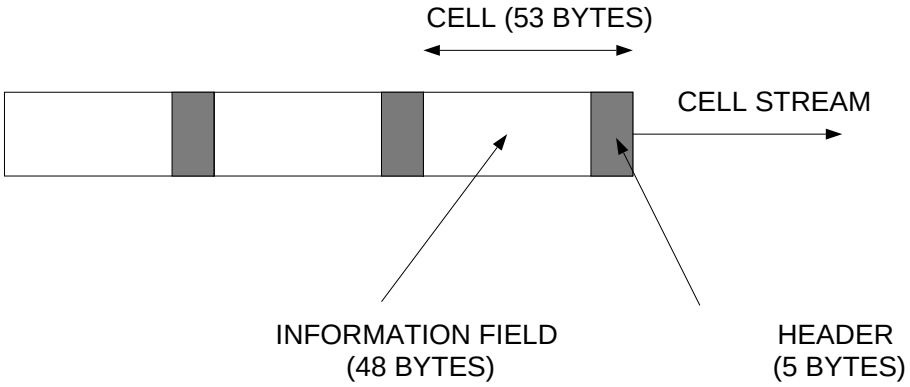


Fig. 2.1. The basic format for an ATM cell. Each cell consists of a 5 byte header and a 48 byte information field. The header contains routing information and a byte of code used to detect and possibly correct the occurrence of error in the header. The information field can consist of user data or network data.

into one cell flow. In the receive direction, the cell demultiplexing function directs individual cells to the appropriate virtual path and virtual channel.

When a cell arrives at a switching node, the node uses the virtual path identifier, the virtual channel identifier, and routing information established at connection setup, to determine the outgoing link on which to send the cell. The switching element may also change the virtual path identifier and virtual channel identifier values to new ones that are to be used at the next node the cell reaches. Having reached its destination the cell's header is stripped off at the terminating ATM layer and the user payload is transported to the layer above. The reverse process is performed at the transmitting end, that is, the ATM layer appends the header to the user payload prior to transmission.

Cell flow towards the network is regulated by the generic flow control (GFC) function.

Figure 2.2 depicts the header structures at a user-network interface (UNI) and a network-node interface (NNI). The payload type (PT) field is used to indicate whether the cell payload contains user information or network information. The cell loss priority bit (CLP) may be set by the user or the service provider to indicate lower priority cells. Such cells run the risk of being discarded, depending on the conditions of the network. To detect bit errors in the header, the header error control (HEC) field is processed by the physical layer. These eight bits are used for single bit error correction or multiple bit error detection.

2.2 ATM Adaptation Layer (AAL)

The ATM Adaptation Layer is designed to support different types of applications and different types of traffic. The variety of traffic it must accommodate ranges from connection less and asynchronous type, such as file transfers, to connection-oriented and synchronous traffic, such as voice and video.

The AAL is divided into two sublayers: the convergence sublayer (CS) and the segmentation and reassembly sublayer (SAR). The SAR layer is concerned with the

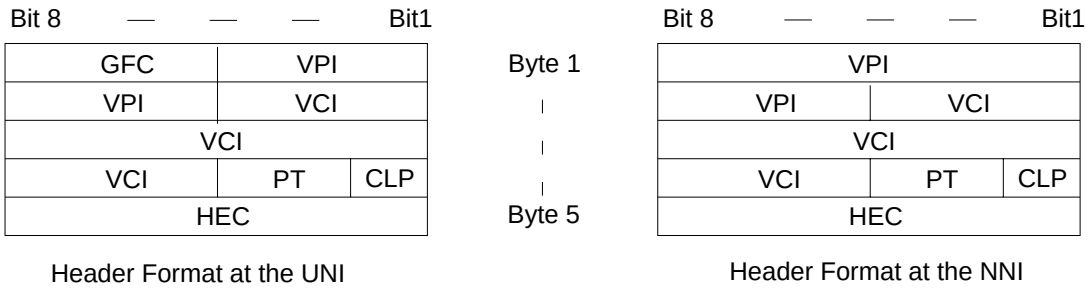


Fig. 2.2. Format of the cell header at both the user-network interface (UNI) and the network-node interface (NNI). At the UNI, the header contains 4 bits used by the generic flow control (GFC) function for regulating data flow towards the network, an 8 bit virtual path identifier (VPI), a 16 bit virtual channel identifier (VCI), 3 bits to indicate the type of payload, 1 bit to indicate the priority of the cell, and 8 bits for detecting multi bit errors and correcting one bit errors within the header. The GFC field is stripped off once the cell enters the network and the VPI extended to 12 bits.

segmentation of higher layer information into a suitable size for the information field of an ATM cell. Its task is to also reassemble the contents of ATM cell information fields into higher layer information. Functions such as processing of cell delay variation, end-to-end synchronization, and handling of loss and misinserted cells are performed by the CS sublayer.

As shown in Figure 2.3, the ATM Adaptation Layer accepts user data, which could range in size from one byte to several thousand bytes, and places a header and trailer around it. The resulting data unit is segmented into smaller data units ranging in size from 47 to 44 bytes according to the type of traffic. Subsequently, a header and trailer are added to each data unit. The nature of the header and possible use of a trailer will vary depending on the type of payload that is being supported. In any event the final data unit will always be a 48 byte protocol data unit (PDU).

All functions specific to the services are provided at the boundary of the ATM network and are performed by the ATM Adaptation Layer. The user payload is carried transparently by the the ATM network. The network does not process the user payload nor does it know the structure of the data unit. The functions performed in the ATM Adaptation Layer depend upon the higher layer requirements, hence, it must support multiple protocols to fit the needs of its different service users. In order to minimize the number of different protocols, the services are classified according to the timing relationship between source and destination, bit rate, and connection mode.

2.2.1 Physical Layer

The physical layer consists of two sublayers: the physical medium (PM) sublayer and the transmission convergence (TC) sublayer. The PM sublayer includes only physical-medium dependent functions and provides bit transmission capability, including bit transfer and bit alignment. The TC sublayer performs cell rate decoupling, HEC generation/verification, cell delineation, transmission frame adaptation, and transmission frame generation and recovery.

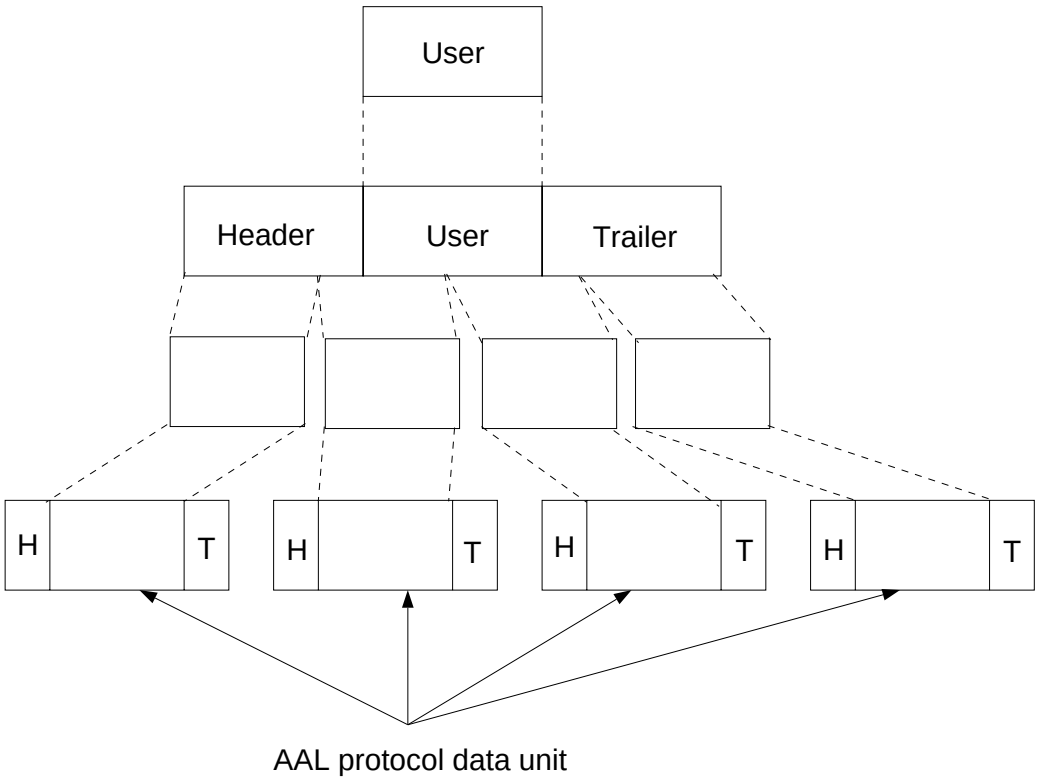


Fig. 2.3. ATM Adaptation Layer convergence, and segmentation and reassembly sublayers. The convergence sublayer places a header and trailer around the user data before passing it onto to the segmentation and reassembly sublayer. The segmentation and reassembly sublayer segments the entire data unit and places headers and trailers around the segments such that each segment, header and trailer are 48 bytes long.

2.3 ATM and SONET

SONET (Synchronous Optical Network) is an optical based carrier transport network utilizing synchronous operations between components [10]. It can support all types of traffic and is based on optical fiber technology. It is used to provide the physical layer for ATM networks. Incoming 51.84 Mbits/s STS (synchronous Transport Signals) streams are multiplexed onto optical carrier signals, that are designated by OC-n, where n is the multiplex integer. Most current implementations of ATM use OC-3 and hence are routed at 155 Mbits/s.

2.4 The AAL, MPEG, and Cell Concealment

We describe the syntax of the MPEG-1 bit stream in the next chapter. An MPEG-1 sequence contains data crucial to its proper decoding. Thus, whenever this information is to be packed into a cell, the CLP bit will have to be set to zero to protect that part of the MPEG sequence. This means that the syntax of the bit stream must be known. Thus, the switching element at the user-network will have to decode the sequence to determine the crucial bits to protect them. This can only be done at the ATM Adaptation Layer, and in particular at the convergence sublayer. The AAL will then convey to the ATM Layer which data units are to be protected. In addition since our packing scheme, described in Chapter 6, requires that extra bits be inserted in each cell, this will similarly have to be done at the convergence sublayer.

3. MPEG VIDEO COMPRESSION STANDARD

Video has traditionally been recorded, stored and transmitted in analog form. However, digital video has rapidly gained popularity [13, 14, 15]. Advances in digital video technology and storage have made it possible to integrate digital video into a number of multimedia applications. In response to a growing demand for a common format for coding and storing digital video, the International Organization for Standardization (ISO) established the Moving Pictures Expert Group (MPEG) to develop standards for coded representations of moving pictures and associated audio, for storage on digital media. MPEG, formally known as group
ISO/IEC JTC 1/SC 29/WG 11, developed the ISO/IEC 1172 standard [16]. This standard, commonly known as MPEG-1, consists of 4 parts. Part 1, the systems standard, specifies the system coding layer. It defines a multiplexed structure for combining audio and video data, and the means for representing the timing information needed to replay synchronized sequences in real time. In Part 2, the video standard, the coded representation of video data and the decoding process required to reconstruct pictures are specified. The coded representation of audio data and the decoding process required to reconstruct audio signal are specified in Part 3, the audio standard. Part 4, the compliance testing standard, specifies the procedures for determining the characteristics of coded bit streams and the decoding process. It also specifies the procedures for testing compliance with the requirements stated in the other parts of the standard. MPEG also developed two another standards, MPEG-2 and MPEG-4, and is in the process of developing another standard, MPEG-7, for describing various types of multimedia information.

MPEG-2, formally known as ISO/IEC 13818, is used for High Definition Television and provides a suite of algorithms for generating hiqh quality video at various resolutions and data rates. MPEG-4, formally known as ISO/IEC 14496, is however, a standard for multimedia applications. It provides provides a set of technologies that satisfy the needs of authors, service providers and end users. MPEG-4 enables the production of content that has far greater reusability and greater flexibility than is possible today with individual technologies such as digital television, animated graphics, World Wide Web (WWW) pages and their extensions. It also offers transparent information, which can be interpreted and translated into the appropriate native signaling messages of each network with the help of relevant standards bodies. Furthermore, it brings higher levels of interaction with content, within the limits set by the author[66].

Other standards have also been developed by the International Telecommunications Union-Telecommunications Standardization Sector (ITU-T). These are the ITU-T H.263 and ITU-T H.261 video conferencing standards, and are similar to MPEG-1.

3.1 CCIR601 Digital Video

In 1940, the Federal Communications Commission (FCC) established the National Television Systems Committee (NTSC) to develop transmission standards for television signals. The committee first developed the standard for televising monochrome signals, and in 1953 their standard for color television systems was approved by the FCC. Broadcasting of color television began in 1954.

To transmit two dimensional information, such as an image, some form of scanning is required. In television, scanning is performed as shown in Figure 3.1, where a scanning spot starts at A and scans the image till it reaches B. At that instant it is blanked out. It then traverses across the width of the image to C, where it is turned on. From C it continues to scan the image along the solid lines, being turned off at the right edge of the image and on at the left, till it reaches D. At D it is turned off

and proceeds to vertically retrace, the image back to E, but at a much faster rate. The scanning process is repeated, this time along the broken lines, until the spot reaches H. From there it proceeds back to A to repeat the whole process. This form of scanning is called interlaced scanning. The two sets of lines are called fields and together they constitute a frame. The scanning rate is rapid enough to create an illusion of continuous motion, and the field rate makes flickering imperceptible to the human eye.

According to the NTSC standard, the scanning rate for color signals is 29.97 frames/s. Each frame consists of a total of 525 lines, 480 of which contain actual information. The resulting analog signal is decomposed into its red (R), green (G), and blue (B) color components. The RGB components are transformed into the YIQ [17, 18] color space by means of the following transformation:

$$\begin{bmatrix} \mathrm{Y}' \\ \mathrm{I}' \\ \mathrm{Q}' \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} \mathrm{R}' \\ \mathrm{G}' \\ \mathrm{B}' \end{bmatrix}$$

where R′, G′, and B′ are the gamma corrected R, G, and B components, respectively, and Y′, I′, and Q′ are the corresponding Y, I, and Q components, respectively. The inverse of this transformation is performed at the receiving end to produce gamma corrected R, G, and B signals. The need for having gamma corrected signals is due to the fact that the intensity of light reproduced at the screen of a Cathode Ray Tube (CRT) is a non linear function of its input voltage [19]. This non linearity can be expressed in the form $I = V^{\gamma}$, where I is the intensity and V the applied voltage. Thus to obtain a linear relation between I and V, the voltage applied should be $V^{\frac{1}{\gamma}}$. This preprocessing of the signal is called gamma correction.

The bandwidth of each of the Y′, I′, and Q′ signals is approximately 4.2 MHz. At the transmitter the I′, and Q′ signals are band limited to 1.5 MHz and 0.6 MHz, respectively, and are quadrature modulated on a 3.579 MHz chroma sub carrier. The resulting signal is the NTSC composite baseband signal. Denoting this signal by

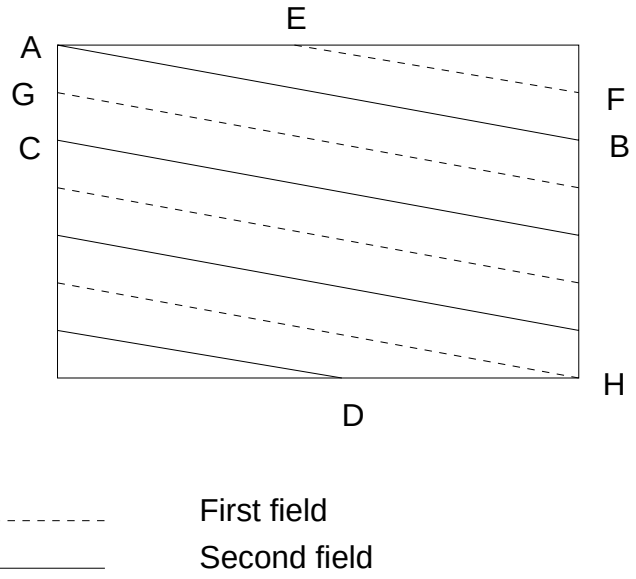


Fig. 3.1. Interlaced scanning. Each image is first scanned along the solid lines and then along the broken ones. Each set of lines constitute a field, and both fields make up a frame. The scanning rate, according to the NTSC standard, is 30 frames/s consisting of 525 lines/frame. In the European standards the scanning rate is 625 lines/frame at 25 frames/s.

$x_{NTSC}(t)$, and the Y′, I′, and Q′ signals by $x_Y(t)$, $x_I(t)$, and $x_Q(t)$ respectively, then

$$x_{NTSC}(t) = x_Y(t) + x_I(t)cos(2\pi f_{cc}t) + x_Q(t)sin(2\pi f_{cc}t)$$

where f_{cc} is the chroma sub carrier frequency. In Europe, standards that use 25 frames/s and 625 lines/frame [17] have been developed.

When it is desired to digitize a video signal, should the NTSC composite signal be used or should each color component be digitized separately? Both approaches have been used. In the area of video compression, component signals are usually used. When each line is sampled at the rate of 13.5 MHz this yields 720×480 active samples per color component [14].

To foster progress in digital video, the International Telecommunications Union (ITU), formerly known as the International Radio Consultive Committee (CCIR), defined standards, namely Recommendation CCIR601 [20], for the digital coding of television signals in component form. In broad terms, a component digital video signal is obtained by decomposing an analog video signal into three color components, sampling each color component and quantizing the samples [13, 14, 15]. This results in three separate discrete space-discrete amplitude signals or pictures.

A different color space is used in coding component digital video, this is the YC_rC_b space. In converting an RGB signal to the YC_rC_b color space, the luminance component, Y, and color differences $(R-Y)$ and $(B-Y)$ are first obtained according to the following transformation:

$$\begin{bmatrix} Y \\ (R-Y) \\ (B-Y) \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.701 & -0.587 & -0.114 \\ 0.299 & -0.587 & 0.886 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

As a result of the above transformation, Y assumes values in the range [0,1.0], that is, it is normalized, whereas $(R-Y)$ and $(B-Y)$ take on values in the ranges [-0.701,0.701], and [-0.886,0.886]. Their normalized values, denoted by, C_R and C_B, respectively, are obtained as follows:

$$C_R = \frac{0.5}{0.701}(R-Y)$$

$$\mathrm{C_B} = \frac{0.5}{0.886}(\mathrm{B} - Y)$$

All normalized components are scaled and shifted in value. Denoting the resulting components by $\overline{Y}$, $\overline{C}_R$, and $\overline{C}_B$, then

$$\overline{Y} = 219Y + 16$$

$$\overline{C}_R = 224\mathrm{C_R} + 128$$

$$\overline{C}_B = 224\mathrm{C_B} + 128$$

The Y, $\mathrm{C_r}$, and $\mathrm{C_b}$ values are then the 8 bit quantized versions of $\overline{Y}$, $\overline{C}_R$, and $\overline{C}_B$. This results in the Y component being quantized to 220 levels, and the $\mathrm{C_r}$ and $\mathrm{C_b}$ components to 255 levels. The minimum Y value, the black value, is 16, whereas the minimum $\mathrm{C_r}$ or $\mathrm{C_b}$ value is -128. If the RGB signal had already been quantized, then the $\mathrm{YC_rC_b}$ components are obtained by means of the following transformation:

$$\begin{bmatrix} \mathrm{Y} \\ \mathrm{C_r} \\ \mathrm{C_b} \end{bmatrix} = \begin{bmatrix} \frac{77}{256} & \frac{150}{256} & \frac{29}{256} \\ \frac{131}{256} & -\frac{110}{256} & -\frac{21}{256} \\ -\frac{44}{256} & -\frac{87}{256} & \frac{131}{256} \end{bmatrix} \begin{bmatrix} \mathrm{R} \\ \mathrm{G} \\ \mathrm{B} \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix}$$

All RGB signals have to be gamma corrected prior to their conversion to the $\mathrm{YC_rC_b}$ space.

CCIR601 defines what is known as the 4:2:2 standard, in which the $\mathrm{C_r}$ and $\mathrm{C_b}$ components are subsampled by a factor of two in the horizontal direction, relative to the Y component. Sampling component digital video signals is performed in an orthogonal and progressive manner, top to bottom and left to right. The luminance signal is sampled at 13.5 MHz, while the color difference signals are sampled at 6.75 MHz. This yields 720×480 active samples in the luminance component, and 360×480 active samples in each of the color components. Another format, known as 4:1:1 also exists. In this case, the $\mathrm{C_r}$ and $\mathrm{C_b}$ components are subsampled by a factor of two in the horizontal and vertical directions relative to the Y component. This thus results in 720×480 Y samples, and 360×240 $\mathrm{C_r}$ and $\mathrm{C_b}$ samples.

The H.261 standard defines two non interlaced formats, known as the common interchange format (CIF) and quarter-CIF (Q-CIF) format. In both cases color pictures are represented by their Y, C_r, and C_b components, where each C_r and C_b component contains half the number of lines and pixels of the Y component. In CIF images the Y component consists of either 360/352 pixels per line at 240 non interlaced lines per picture for 30 pictures per second sequences, or of 360/352 pixels per line at 288 non interlaced lines per picture for 25 pictures per second sequences. In the Q-CIF format, which was defined for low bit-rate applications, each image has half the number of lines and pixels of a CIF image. A 4:1:1 CIF sequence has roughly the same quality as the signal recorded on a VHS video tape.

Other formats have been described by the Grand Alliance for High Definition Television (HDTV) [21]. One of the formats set, that applies for both interlaced and non interlaced sequences, requires images to be 4:2:2 YC_rC_b. The Y component of each frame is to have 1920 pixels/line and 1080 lines/frame.

The data rates for uncompressed 4:2:2 Grand Alliance, 4:2:2 CCIR601, 4:1:1 CIF and 4:1:1 Q-CIF video sequences are:

4:2:2 Grand Alliance: $1920 \times 1080 \times 30 \times 8 + 960 \times 1080 \times 30 \times 8 \times 2 = 995.3$ Mbits/s

4:2:2 CCIR601: $720 \times 480 \times 30 \times 8 + 360 \times 480 \times 30 \times 8 \times 2 = 165.8$ Mbits/s

4:1:1 CIF: $360 \times 240 \times 30 \times 8 + 180 \times 120 \times 30 \times 8 \times 2 = 31.1$ Mbits/s

4:1:1 Q-CIF: $180 \times 120 \times 30 \times 8 + 90 \times 60 \times 30 \times 8 \times 2 = 7.8$ Mbits/s

It is evident that Ethernet channels (10 Mbits/s) are capable of transmitting 4:1:1 Q-CIF sequences only. A DS-1 channel (1.5 Mbits/s), however would not be able to support any of the above uncompressed signals, and an OC-3 (155.5 Mbits/s) and a DS-3 (45 Mbits/s) link would only be capable of supporting 4:1:1 CIF and 4:1:1 Q-CIF sequences. In fact, it would require 5 OC-3 links to support a 4:2:2 Grand Alliance digital video signal, and 4 DS-3 lines to support a 4:2:2 CCIR601 video sequence. In light of the above, it is necessary to develop compression schemes to reduce these data rates.

3.2 MPEG-1 Basics

The main objective of the MPEG-1 standard was to compress 4:1:1 CIF sequences (31.1 Mbits/s) to a target bit rate of 1.5 Mbits/s. This required a 30:1 compression ratio, which was to be achieved while providing such features as random access, fast forward, or fast and normal reverse playback [22]. The standard defines a generic decoder but leaves the implementation of the encoder open to individual designs.

Digital video is compressed by reducing the temporal and spatial redundancies in every frame. The standard defines three types of frames, depending on the techniques utilized to compress them. They are the *intracoded* (I), *predicted* (P), and *bidirectionally-predicted* (B) Pictures. I pictures are compressed by reducing their spatial redundancies, while P and B pictures are compressed by reducing their temporal redundancies. P pictures are encoded using motion compensated prediction from a past I or P picture, as shown in Figure 3.2. B pictures, however, require both past and future reference frames (I or P pictures) for motion compensation, as illustrated in Figure 3.3. The outcome is that I pictures have the highest data rate and lowest motion artifacts, whereas B pictures have the lowest data rate and highest motion artifacts. Typical data rates are 1 bit per pixel, 0.1 bits per pixel, and 0.015 bits per pixel for I, P, and B pictures respectively.

3.3 MPEG-1 Structure

The image color space used in MPEG-1 is the 4:1:1 YC_rC_b space. Each video frame is divided into non-overlapping regions of pixels, known as macroblocks, each of which contains 16×16 pixels of the luminance component, 8×8 pixels of the C_b chrominance component and 8×8 pixels of the C_r chrominance component. An 8×8 pixel array is known as a block. Thus a macroblock consists of 4 luminance blocks and 2 chrominance blocks (one C_b block and one C_r block). An integral number of macroblocks, arranged in lexicographic order, are grouped together to from a slice. Slices within a picture can be of different sizes, as shown in Figure 3.4. The division

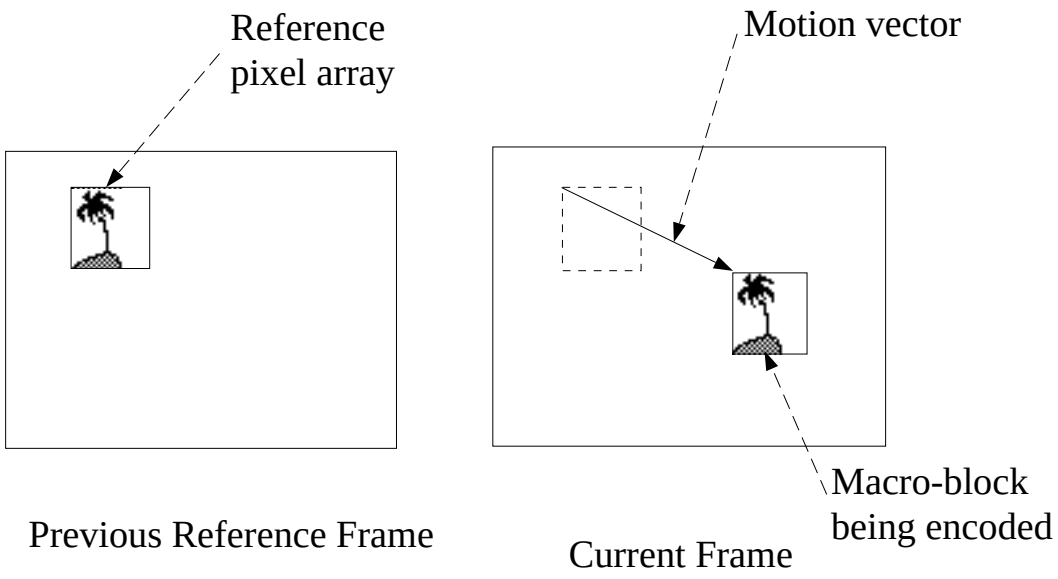


Fig. 3.2. Motion compensated prediction using past frames. Macroblocks in the current frame are compared to macroblock sized regions in the previous frame for a close match up. The displacement between them is the motion vector, which is encoded rather than the DCT coefficients.

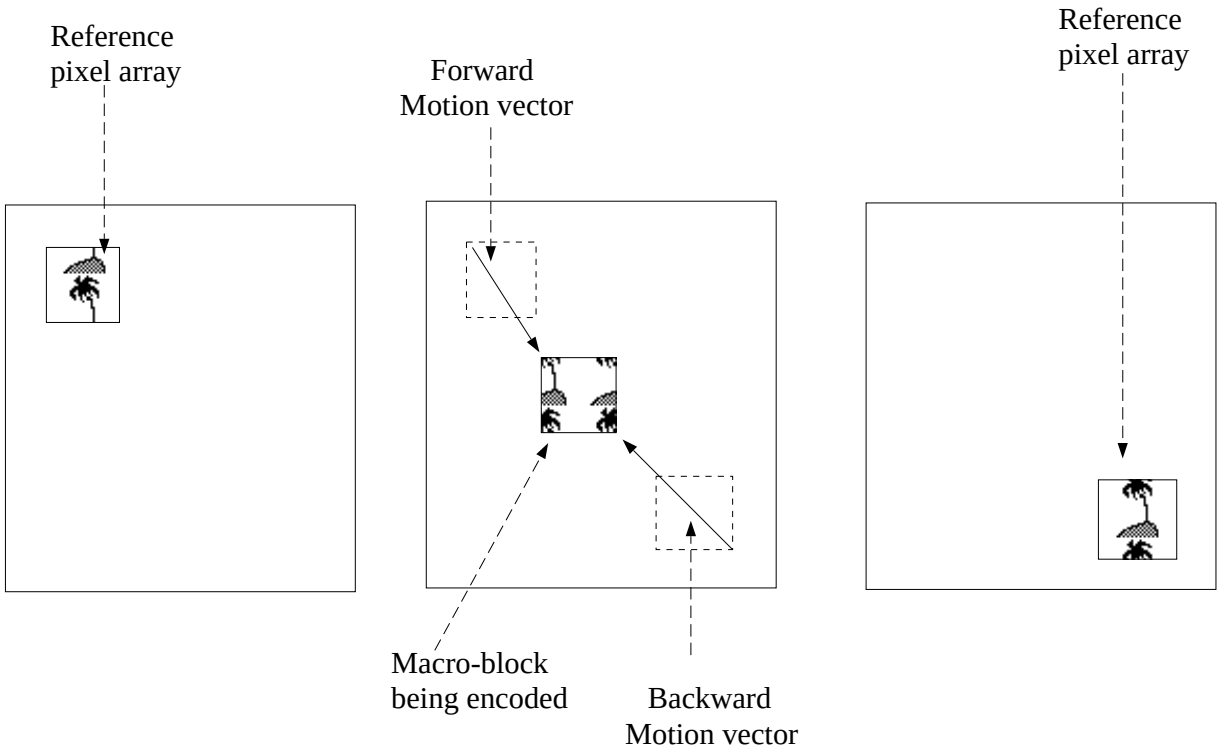


Fig. 3.3. Motion compensated interpolation using past and/or future frames. A macroblock in the current frame is compared to macroblock sized regions in a previous frame and a future frame, for a close match up. The average of both closely matching regions is obtained. The displacements between the current macroblock and the matching regions in the previous and future frames are the forward and backward motion vectors, respectively. If the average of both regions is the most closely matching, then both motion vectors are coded. Otherwise, the motion vector pointing to the most closely matching region is coded.

of one picture into slices need not be the same as the division of any other picture. A slice can begin and end at any macroblock in a picture provided that the first slice begin at the top left of the picture, and the end of the last slice be the bottom right macroblock of the picture. There can be no gap between slices, nor can slices overlap. The minimum number of slices in a picture is one, and the maximum number is equal to the number of macroblocks. Several consecutive pictures, arranged in display order, are combined to form a structure known as a group of pictures (GOP), as shown in Figure 3.5. A GOP must contain at least one I picture, which may be followed by any number of I and P pictures. Any number of B pictures may interspersed between each pair of I or P pictures, and may also precede the first I picture. Since decoding B pictures requires both past and future pictures, the ordering of the pictures is changed prior to the transmission or storage of the encoded bit stream. The reference I and P pictures are placed before their dependent B pictures. GOPs facilitate the implementation of features such as random access, fast forward, or fast and normal reverse playback [22].

3.4 Intracoding

To generate I pictures, the two dimensional discrete cosine transform (DCT) [23] of each 8×8 block is obtained. This is done for all three color components. Each array of 8×8 DCT coefficients is then quantized and coded [24]. Normally the number of non-zero quantized coefficients is quite small. This is one of the main reasons why the compression scheme is effective.

Quantization is performed by dividing each coefficient by a quantizer step size and rounding to the nearest whole number to produce the quantized coefficients. Quantization step sizes can be used to control the data rate at the output of the encoder, in which case an increase in a quantizer step size will result in a decrease in the output data rate [25]. After quantization all coefficients are arranged in a zig zag order, as shown in Figure 3.6 and then variable length coded.

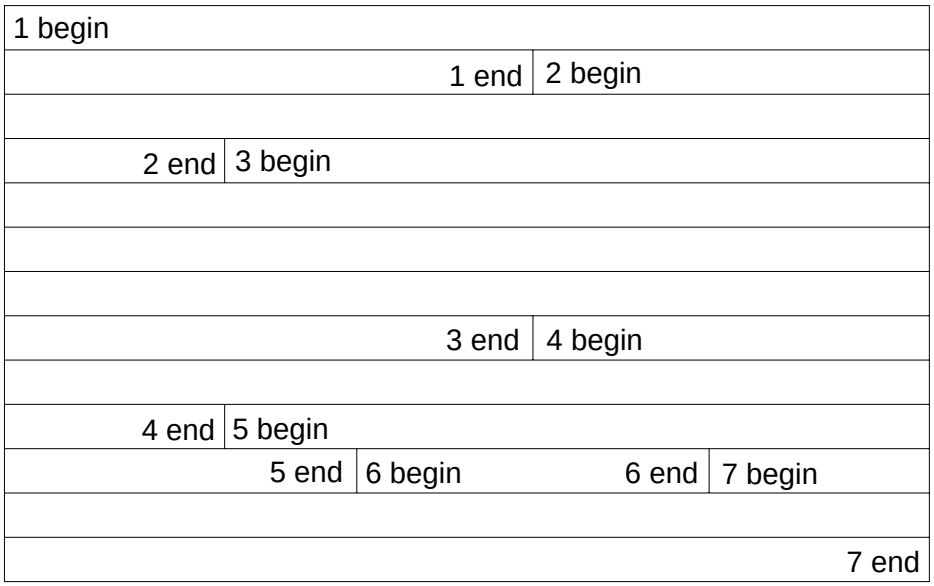


Fig. 3.4. Possible slice configuration in a picture. Slices contain an integral number of macroblocks, and can be of different sizes. A slice can begin and end at any macroblock in a picture provided that the first slice begins at the top left of the picture, and the end of the last slice be the bottom right macroblock of the picture. There can be no gap between slices, nor can slices overlap.

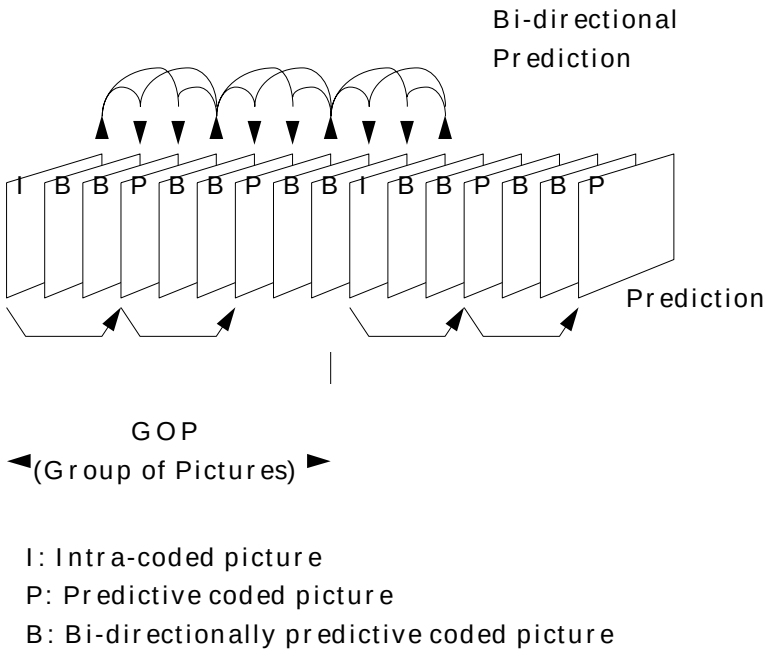


Fig. 3.5. Grouping of pictures into GOPs. A GOP must contain at least one I picture, which may be followed by any number of I and P pictures. Any number of B pictures may interspersed between each pair of I or P pictures, and may also precede the first I picture.

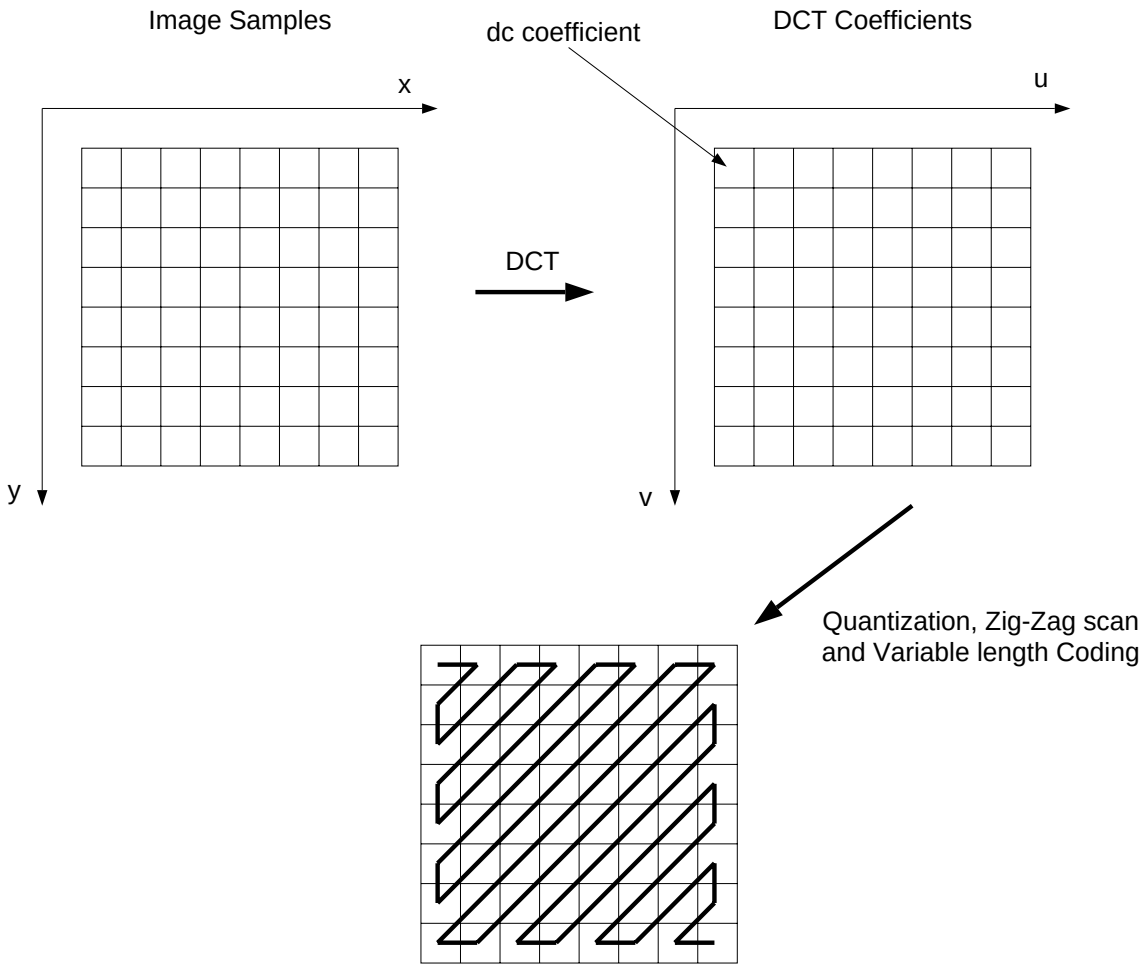


Fig. 3.6. Transform coding, quantization, and run length coding. The DCT of each 8×8 block is obtained. The coefficients are quantized, arranged in zig zag scan, and variable length coded.

At the receiving end the decoder reverses the zig zag scan, decodes the coded coefficient values, multiplies the coefficients by the quantizer step sizes, and performs the Inverse DCT (IDCT) [23] to obtain the pixel values. It is to be noted that due to quantization the resulting pixel values will differ from the original values.

3.5 Motion Compensated Prediction

Motion compensation is used to reduce temporal redundancy. This step is the most computationally intensive step in MPEG. Motion compensated techniques are either predictive or interpolative. Motion compensated prediction assumes that a macroblock in the current frame or picture can be modeled as the translation of a macroblock sized region in a reference picture at some previous or future time, as shown in Figure 3.7. A search range is set up, and a search conducted for that region that best matches the current macroblock. A match is found when a certain criterion or cost function, such as minimum mean square error (MMSE) or minimum mean absolute difference (MMAD), is minimized. The displacement between the centers of the macroblock and the best matching region in the reference frame is the motion vector [26]. When performing the matching process, macroblocks can be displaced by either integer or half pixel displacements. The choice of the matching criterion is left open to the individual designs

Many methods are used to search for motion vectors. These include full search and logarithmic search [15]. The standard does not define which technique is to be used. The simplest search method is the full or exhaustive search scheme. In this case, the cost function is evaluated for every possible displacement. The region giving rise to the minimum of all the obtained cost functions is chosen. Full search, however, is computationally expensive. Other suboptimal search strategies [26, 27], that minimize computation and yield reasonable matches can be implemented.

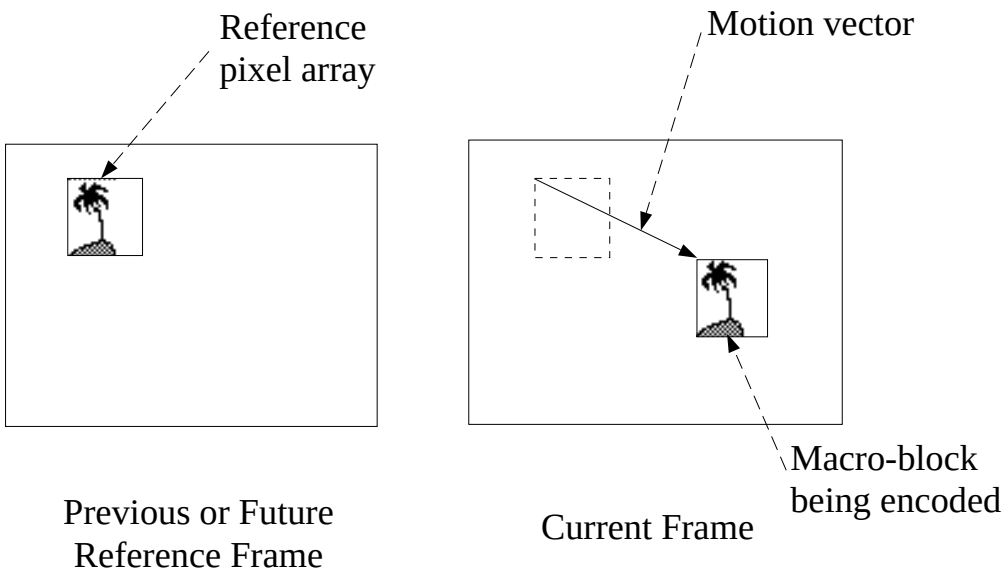


Fig. 3.7. Motion compensated prediction using past or future frames. A macroblock in the current frame is compared to macroblock sized regions in a previous frame and a future frame, for a close match up. The displacements between the current macroblock and the matching regions in the previous and future frames are the forward and backward motion vectors, respectively. The motion vector pointing to the most closely matching region is coded.

3.6 Predictive and Interpolative Coding

To code P and B pictures, the encoder performs motion compensation for each macroblock. In the case of P pictures, past I or P frames are used as reference frames, whereas macroblocks in B frames are compared to macroblocks in past as well as future I or P frames [22]. If no match can be found for a particular macroblock, or if the minimized criteria is greater than a certain threshold, then that macroblock is intracoded. If however, the minimum value is less than the first threshold but greater than a second threshold, the motion vector is variable length coded and the difference between the match and the original intracoded. If the minimum value is less than both thresholds, then only the motion vector is variable length coded.

3.7 MPEG-1 Video Bit Stream Structure

An MPEG-1 coded bit stream commences with a sequence start code followed by a sequence header, data belonging to one or more groups of pictures (GOPs), and a sequence end code. Both the sequence start code and the sequence end code are unique 32 bit integers. The sequence header consists of fields pertaining to the size of the coded frames, the pixel aspect ratio, the picture rate, the bit rate, and the decoder buffer size [16].

Data belonging to a GOP commences with a 32 bit unique start code followed by a GOP header and the data belonging to each picture. The GOP header includes a time code used to provide the time in hours, minutes, and seconds of the sequence, as well as whether or not the frame rate is 30 Hz. Picture data consists of a 32 bit unique picture start code, a picture header, and data belonging to the slices in each picture. The picture header contains a information regarding the type of the picture and motion vector search ranges, as well as a temporal reference [16]. The temporal reference is used to keep count of the number of pictures in the sequence, as well as the order in which they are displayed.

The data belonging to each slice in every picture consists of a 32 bit unique slice start code, a slice header, and macroblock data. The slice start code is a 32 bit code that provides the address of the slice. The slice header contains a quantizer scale used to scale the quantization step sizes [16].

Macroblock data consists of an optional stuffing field used by the encoder to prevent underflow, macroblock addressing data, macroblock type, coded motion vectors, optional quantizer scale, and the data of each block belonging to the macroblock. The macroblock addressing data is the address of the macroblock relative to that of the preceding macroblock. The first macroblock in a slice is addressed relative to the slice to which it belongs, and relative to the last macroblock in the previous row. Block data is the coded quantized DCT coefficients.

All the information contained in the sequence, GOP, picture and slice headers, as well as all the start codes must be safeguarded against cell loss. Any damage incurred to this data can inhibit the decoding process.

3.8 MPEG-2 Video

MPEG-2 is the result of the second phase of the work done by MPEG. It was originally intended to be applicable to a wide class of applications, as well as support a compressed bit rate of 5 Mbits/s for 4:2:2 CCIR601 digital video. Among the original requirements were compatibility with MPEG-1, good picture quality, flexibility of input format, random access capability, fast forward capability, reverse play capability, resilience to bit errors, and bit stream scalability [28]. Bit stream scalability is defined as the ability to discard a portion of the bit stream without compromising the quality of the decoded sequence. Although MPEG-2 can be applied to a wide range of applications ranging from low bit rate to high bit rate, low resolution to high resolution, and low picture quality to high picture quality, yet it would be too complex to attempt to integrate the needs of all applications into a single syntax. Hence, only a limited number of subsets of the syntax are represented by means of profiles and levels [27, 28, 29, 30]. A profile is subset of the bitstream syntax, and

a level is a set of constraints imposed on the values of the parameters of the profile. The standard defines five distinct profiles: Simple, Main, SNR (signal-to-noise ratio) scalable, Spatially scalable and High. The levels used to constrain the profiles are Simple (for CIF resolution pictures), Main (for 4:2:2 CCIR601 resolution pictures), High-1440 and High (for HDTV resolution pictures) [28, 29, 30]. Table 3.1 depicts the upper limits on picture size, frame rate, bit rate, and buffer size for the defined levels.

The full MPEG-2 syntax is divided into two major categories: the non scalable syntax, of which MPEG-1 syntax is a subset, and the scalable syntax. The main feature of the non scalable syntax is the added tools for compressing interlaced video, and that of the scalable syntax is the ability to reconstruct video from partial information.

3.8.1 Non Scalable Syntax

In addition to supporting the coding of progressive video, the non scalable syntax supports the compression of interlaced as well as 4:2:2 video. When coding 4:2:2 frames, two extra C_r and C_b blocks are included within the macroblock structure making macroblocks a total of 8 blocks as opposed to 6 in the case of MPEG-1.

The standard defines three coded frame types, namely I frames, P frames and B frames, and two picture types known as field pictures and frame pictures. A field picture consists of either of the fields belonging to a video frame, whereas a frame picture consists of two interleaved fields. Furthermore, frame or field encoding can be selected on a frame by frame basis. A coded I frame, can be a pair of I field pictures, an intracoded frame, or an I field picture followed by a P field picture. If the latter configuration is used, then the P field picture shall be predicted from the I field picture. Coded P frames are either a pair of P field pictures or a P frame picture. When using two P field pictures, the first field picture shall be predicted from the two most recently decoded reference fields. The second field picture of the current frame is then predicted from the first field of the current frame and the second field of the previous frames. B frames can be a pair of B field pictures, a P field picture followed

by a backward predicted B picture, or a B frame picture. The B field pictures are reconstructed from the two fields belonging to the two most recently decoded reference frames.

Another added feature to MPEG-2 and not available in MPEG-1 is the fact that macroblocks can be split into two 16×8 sub macroblocks, with each sub macroblock having an associated motion vector. Thus predicted macroblocks can have two associated motion vectors and interpolated macroblocks can have up to four associated motion vectors. Furthermore, an intracoded macroblock can have an associated motion vector, known as a concealment motion vector. Although the concealment motion vector plays no role in the coding of the intracoded macroblock, it is used by the decoder to reconstruct the macroblock in the event that it is lost during the transmission of the coded bitstream over lossy channels.

The two non scalable profiles are Simple and Main. The Simple profile does not employ backward prediction in the coding process, that is no B pictures are coded. Consequently, no picture reordering is required at the decoder. The Main profile, however, supports B pictures, and is capable of decoding MPEG-1 video streams [31, 28, 29, 30]. Both the Simple and Main profile are used to encode 4:1:1 video data and both permit the dc coefficients to have a precision of up to 10 bits, unlike MPEG-1 which only permits a maximum precision of 8 bits .

Table 3.1 Upper bounds on the parameters used in MPEG-2. The numbers enclosed within parentheses are the upper bounds when enhancement layers are used.

Profile	Level	Frame width (pixels)	Frame height (pixels)	Frame rate (frames/sec)	Bit rate (Mbits/sec)
Simple	Main	720	576	30	15
Main	Low	352	288	30	4
	Main	720	576	30	15
	High-1440	1440	1152	60	60
	High	1920	1152	60	80
SNR scalable	Low	352	288	30	3 (4)
	Main	720	576	30	10 (15)
Spatially scalable	High-1440	720 (1440)	576 (1152)	30 (60)	15 (40) (60)
High	Main	352 (720)	288 (576)	30 (30)	4 (15) (20)
	High-1440	720 (1440)	576 (1152)	30 (60)	20 (60) (80)
	High	960 (1920)	576 (1152)	30 (60)	25 (80) (100)

3.8.2 Scalable Syntax

Scalability allows for a layered representation of the coded bit stream. A scalable bitstream can consist of up to three layers, a base layer and two enhancement layers. MPEG-2 syntax allows for four modes of scalability: data partitioning, SNR scalability, spatial scalability, and temporal scalability [27, 28]. With the exception of data partitioning, the base layer can be an MPEG-1 sequence, a bitstream adhering to the Simple profile, or a Main profile bitstream.

In data partitioning, the bit stream is split into two layers, called partitions. The first partition may include all critical header information, and the second may include the remaining bit stream. This would typically be employed when the video bitstream is to be transmitted over a lossy channel. For example, the first partition could be packed into high priority ATM cells, and the the second partition into low priority ATM cells, prior to transmission [28].

SNR scalability can be used in applications that support video transmission at multiple quality levels. All layers have the same spatial resolution but different video qualities. The lower layer is coded by itself and provides the basic video quality. The enhancement layers are used to refine the data for the DCT coefficients of the lower layer. After the DCT coefficients of the lower layer have been quantized, the difference between the original and quantized coefficients is obtained. This difference is then quantized, variable length coded, and transmitted as part of the enhancement layer of the bit stream. If required, this extra data can be retrieved by the decoder for improving the quality of the decompressed frames [28].

The purpose behind spatial scalability is to divide the bit stream into layers of different spatial resolution. At the encoder the sequence is encoded at a lower resolution, also known as the base layer. The base layer is then upsampled and compared to a higher resolution sequence. At the same time, motion compensated prediction is performed at the higher resolution. The encoder then decides to either encode the difference between the upsampled version of the base layer and the high resolution

frame, the difference arising from performing motion compensated prediction, or the difference arising from using a weighted combination of the predicted and upsampled version. The difference is then quantized and variable length coded as the enhancement layer. To reconstruct the sequence the decoder first decodes and reconstructs the base layer. A higher resolution frame is then predicted or interpolated from reference high resolution frames by means of motion vectors, by upsampling the base layer, or by using a weighted combination of both the predicted/interpolated and upsampled versions . The data coded in the enhancement layer is then used to improve the quality of the reconstruction [28, 30]. Similar to the data partitioning case, Spatial scalability, can be employed when the video bitstream is to be transmitted over a lossy channel. The base layer can be packed into high priority ATM cells for safeguarding and the enhancement layers into priority ATM cells, before being sent [28]. This however is not a very practical method if the base layer consisted of 4:1:1 CIF images, as it would result in the network congestion.

Temporal scalability allows the migration from systems with low temporal resolution to systems with higher temporal resolution. In this case, the lower layer is coded by itself and provides the basic temporal rate. The enhancement layers consist of temporal predictions performed with respect to the lower layer. The two layers are combined to generate a stream at full temporal resolution. All layers have the same frame size and chrominance formats but different frame rates [27, 28].

3.9 MPEG-2 Video Bit Stream Structure

The structure of an MPEG-2 video bitstream is similar to that of an MPEG-1 video bitstream, except that extra data fields that indicate scalability, interlaced video, color format, the presence of error concealment motion vectors, and field or frame coding are inserted where required. The presence of all these extra data fields is indicated by the presence of a unique 32 bit integer positioned right after the sequence header. When this 32 bit integer is absent, the bitstream is MPEG-1 compliant and can be passed off to an MPEG-1 decoder [28].

All header information and start codes must be safeguarded against cell loss. Any damage incurred to this data can inhibit the decoding process.

3.10 MPEG-1 Systems Layer

3.10.1 Introduction

When delivering MPEG compressed video and audio bitstreams to a user, certain information such as presentation times or decoding times of pictures and audio frames have to be supplied. This is necessary for synchronized play back. Such information is not included within the compressed audio or compressed video bitstreams, but within what is known as the MPEG System Layer.

The ISO/IEC 11172-1 standard [32, 29], or system layer standard, addresses the problem of combining one or more data streams, whether they are compressed audio, compressed video or private data, into a single stream. The standard defines data fields and specifies semantic constraints on the data stream to enable decoders and/or encoders perform such functions as synchronized presentation of decoded information, the construction of a multiplexed stream, the management of buffers for coded data, and absolute time identification. The standard does not specify the architecture or implementation of encoders or decoders [32, 29].

An encoding system performs the coding of audio and video data, coding of system layer information, and multiplexing. Coding the system layer information includes creating time stamps such as the presentation and decoding time stamps, which are used for the synchronization of audio and video. Other fields such as the system clock reference fields that are utilized in synchronization and buffer management need also be created.

The decoding system parses and demultiplexes multiplexed streams and decodes and presents elementary streams. The specification of the system is written in terms of an idealized reference decoder, known as the system target decoder, whose purpose is to provide a clear, simple model of the decoding system. The System Target Decoder, shown in Figure 3.8, is composed of System, Video and Audio MPEG decoders. It

accepts a multiplexed stream, and relies on the System Decoder to extract timing information as well as demultiplex the stream into elementary streams that are then passed on to their respective decoders. The timing information extracted, is used to synchronize the operations of the Video and Audio decoders.

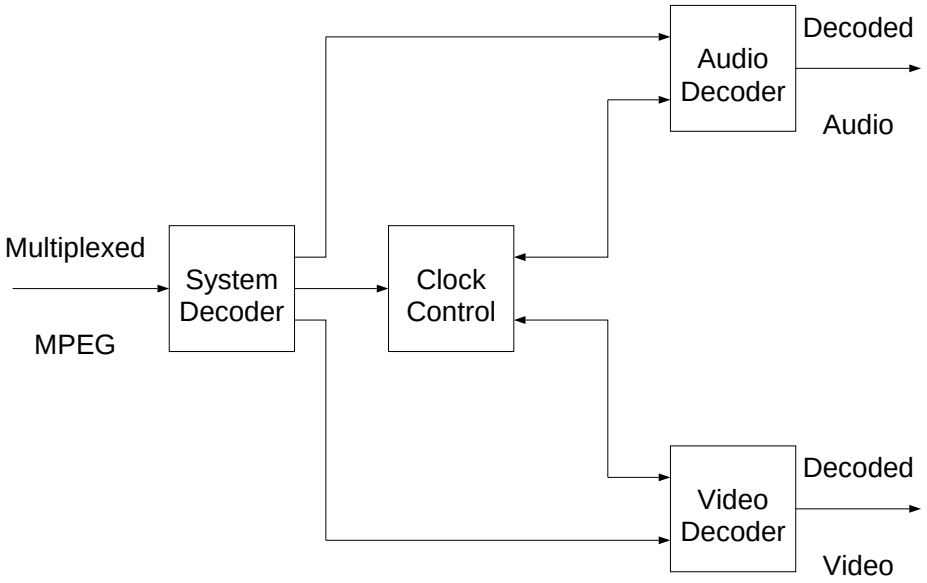


Fig. 3.8. System Target Decoder. The System Target Decoder consists of system, video and audio decoders. The system decoder parses the stream, extracts timing information as well as the video and audio elementary streams. Each elementary stream is then passed onto to its decoder.

3.10.2 Overview of MPEG-1 System Layer

An MPEG-1 multiplexed stream, formally known as an ISO/IEC 11172 stream, consists of one or more packs of data followed by a 32 bit stream end encode, shown in Figure 3.9. According to the standard, packs can have variable length and there must be at least one pack in each stream. Each pack consists of a 12-byte pack header, a system header and an arbitrary number of packets. The system header must be included in the first pack in each multiplexed stream, but need not be included in other packs since all the system headers within a stream contain the same information. If a different system header is to be included, then the stream must be terminated and a new one containing the new system header started. Each packet contains a

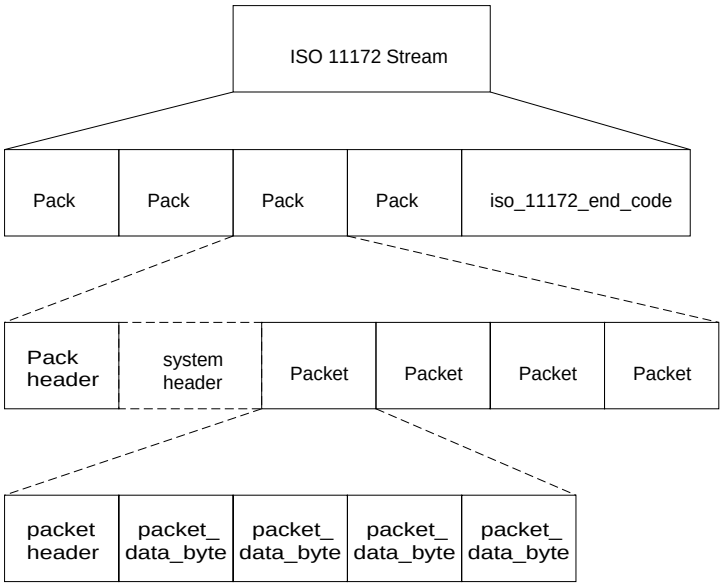


Fig. 3.9. Structure of an MPEG-1 multiplexed stream. The stream consists of variable length packs carrying packets of video or audio data.

packet header followed by video, audio, padding or private streams, represented as packet data [32, 29].

3.10.3 Pack Layer

Each pack starts consists of a unique 32 bit pack start code, followed by the system clock reference (SCR) and the multiplexing rate. The system clock reference is used to indicate the time when the last byte (bits 7 to 0) of its field should enter the system target decoder. The multiplexing rate, specified in units of 50 bytes/sec, indicates the rate at which bytes arrive at the system target decoder. This is followed by a system header, which is optional unless the pack is the first pack in the stream, and an arbitrary number of packets [32, 29].

3.10.4 System Header

The system header is required in the first pack and although allowed in any pack, its values cannot change. It always starts with a unique 32 bit start code. The contents of the system header include a bound on the multiplexing rate, used by

the decoder to assess whether or not it is capable of decoding the entire stream, the number of multiplexed audio and video streams, and bounds on the buffer sizes used by the video and audio decoders.

3.10.5 Packet Layer

A packet always starts with a packet header which is followed by bytes of data from either an audio or video bitstream. The header commences with a 32 bit code that signals both the beginning of a packet as well as the whether the data being carried by the packet is video, audio, or private data. The contents of the packet header also include the buffer sizes of the audio and video decoders as well as the times for decoding and presenting the multiplexed audio and video streams [32, 29].

3.11 MPEG-2 Systems Layer

The MPEG-2 systems layer evolved from the MPEG-1 systems layer. Whereas, MPEG-1 was designed for work with digital storage media that have minimal errors, MPEG-2 systems layer was targeted for work in ATM environments, to handle elementary streams with different time bases, as well as provide backward compatibility with MPEG-1 systems layer. To achieve this two data stream types: the Program Stream and the Transport Stream were defined. Both utilize the same Packetized Elementary video or audio stream structure.

A Packetized Elementary Stream is constructed by packing an audio or video elementary bitstream into a number of packets. Unlike packets in MPEG-1 multiplexed streams, Packetized Elementary Stream packets contain additional fields for optional encryption, packet priority levels, trick mode indications that assist fast forward and slow motion, copyright protection, and packet sequence numbering. In the case of Program Streams, the packets belonging to Packetized Elementary Streams are then multiplexed onto packs that have a structure identical to MPEG-1 system layer packs. This renders a Program Stream decoder capable of decoding MPEG-1

system layer bitstreams [33]. Transport Streams on the other hand have a slightly different structure and will be discussed next section.

According to the standard, Systems, Video, and Audio all have a timing model in which the end-to-end delay from the signal input to an encoder to the signal output from the decoder is a constant. This delay is the sum of encoding, encoder buffering, multiplexing, communication or storage, decoder buffering, decoding, and presentation delays. As part of the timing model all video pictures and audio samples shall be presented once, unless specifically coded to the contrary, and the inter picture arrival and audio sample rate are the same at the decoder and encoder.

All timing is defined in terms of a common system clock, referred to as the System Time Clock. In the Program Stream this clock may have an exactly specified ratio to the video or audio sample clocks, or it may have an operating frequency which differs slightly from the exact ratio, while still providing precise end-to-end timing and clock recovery. In the Transport Stream the system clock frequency is constrained to have the exactly specified ratio to the video and audio sample clocks at all times. This is to simplify sample rate recovery in decoders [33].

3.11.1 Transport Stream

The Transport Stream is a stream definition which is tailored for communicating or storing bitstreams of coded data in environments in which significant errors may occur. Transport Streams may be either fixed or variable rate, and in either case the constituent elementary streams may either be fixed or variable rate. The Transport Stream rate is defined by the values and locations of the Program Clock Reference fields which are analogous to, and serve the same purpose of the System Clock Reference and the multiplexing rate fields of MPEG-1 system layer. A Transport Stream may be constructed from elementary streams, from Program Streams, or from other Transport Streams [33].

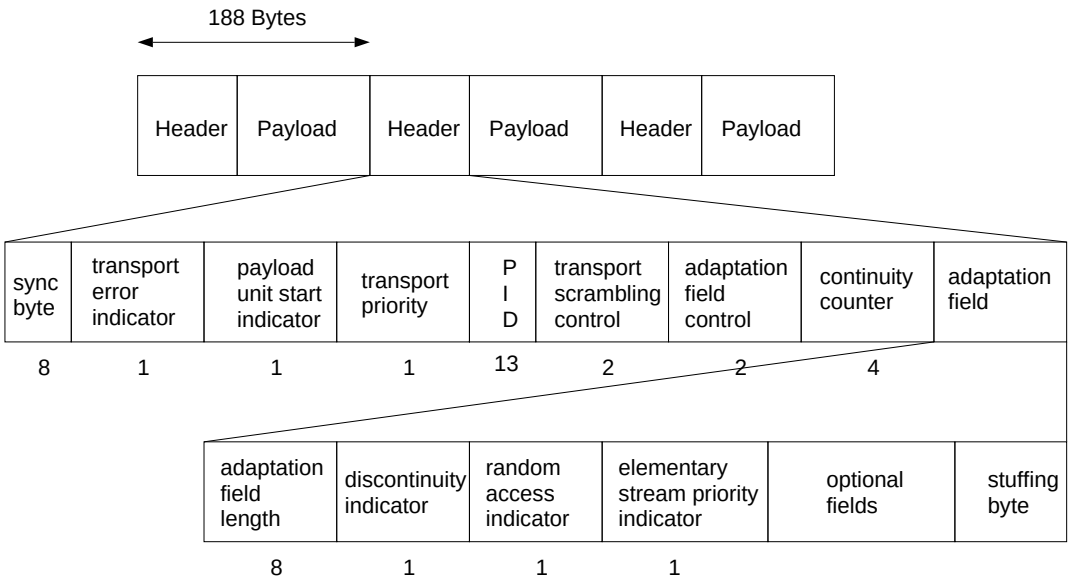


Fig. 3.10. Transport stream packet structure. Each packet is 188 bytes in length and includes a header followed by the multiplexed data. The header contains information needed for proper demultiplexing and synchronized play back.

3.11.2 Transport Stream Syntax

The Transport Stream, as shown in Figure 3.10, consists of a stream of packets, 188 bytes in length, carrying the multiplexed data. Each Transport Stream packet consists of a header section and a payload section. The packet header begins with a 4 byte prefix, which contains a 13 bit packet identifier (PID) field. The packet identifier is used to identify the type of the data carried in the Transport Stream packet payload. All Transport Stream packets carrying the data belonging to the same elementary stream have the same packet identifier [33].

3.11.3 Transport Stream Packet Layer Headers

Each header, as shown in Figure 3.10, consists of a sync byte, which is a '0100 0111' bit string, followed by a transport error indicator flag. The transport error indicator flag, when set to '1', is used to indicate that there is at least one uncorrectable errored bit within the packet. The transport error indicator flag is followed by the payload unit start indicator which is set to '1' if the packet is carrying data from a Packetized

Elementary Stream packet. Furthermore, one and only one Packetized Elementary Stream packet is allowed to commence in the Transport Stream packet payload. The transport error indicator is followed by the transport priority bit which is used to indicate the priority level of the Transport Stream packet. This in turn is followed by 13 bit packet identifier field, the packet encryption field, and a 2 bit field known as the adaptation field control. The values and interpretation of the adaptation field control are given in Table 3.2 below. A Transport Stream packet with a '00' adaptation field control field shall be discarded by decoders.

Table 3.2 Adaptation Field Control Values

Value	Description
00	reserved for future use
01	no adaptation field, payload only
10	adaptation field only , no payload
11	adaptation field followed by payload

A continuity counter field, that is incremented with every non duplicate Transport Stream packet carrying the same packet identifier, follows the adaptation field control. A continuity counter field is said to be continuous if two successive Transport Stream packets having the same packet identifier either have the same continuity counter or their continuity counters differ in value by one. The bytes following the continuity counter are adaptation field bytes, Packetized Elementary Stream data, private data, or stuffing bytes [33].

3.11.4 Adaptation Field

The adaptation field consists, as shown in Figure 3.10, of a length indicator, the discontinuity indicator which is set to '1' whenever there is a discontinuity in the continuity counter, the random access indicator which is used to aid in random access into the Packetized Elementary Stream packets, the priority level indicator of Packetized Elementary Stream packets, optional fields and stuffing bytes.

The optional fields include the Program Clock Reference which serves the same purpose as the System Clock Reference of the MPEG-1 system layer, that is it indicates the time when its last byte should be admitted to the decoder. The Program Clock Reference need not be included within every packet, however it should be periodically updated. Another optional flag following the Program Clock Reference is the splicing point flag which is used if two different elementary streams are concatenated together and packetized within the same Transport Stream packet [33].

With regards to transmitting Transport Stream packets over ATM networks, the field indicating the priority of the Transport Stream packets is of primary interest. It can be used to signal when Transport Stream packets are to be protected by sending them in high priority ATM cells.

4. PREVIOUS WORK IN PACKET LOSS CONCEALMENT

Data networks suffer from packet loss and packet delay jitter. Thus their impact on the decoding process needs to be addressed. For proper decoding of video signals, the encoder and decoder need to be synchronized. The decoder synchronizes its clock to that of the encoder via time stamps that are inserted in the sequence. These stamps, however, are not available in the data carried by every packet. The decoder then has to utilize packet arrival times to derive a stable clock. Packet loss and packet delay jitter greatly impact the synchronization process [34].

Packet delay jitter results in clock instability which manifests itself as image blurring and waving. A receiver buffer is used to lessen the impact of packet delay jitter.

Packet loss has an even greater effect than packet delay jitter. Packet loss greatly offsets the decoder's synchronization, and by the time the decoder has resynchronized itself, major portions of a picture may have been lost. Hence, techniques for circumventing packet loss have to be developed. These methods must be capable of retrieving the undamaged data to decode it, and post processing the the damaged areas for error concealment

Two methods for ameliorating the effect of packet loss, in particular ATM cell loss, were proposed in [8]. It is assumed that both encoding and decoding are occurring simultaneously. It is also assumed once the decoder has detected cell loss, it communicates to the encoder the location of the damaged picture regions. The encoder then decides to do either of two things. The encoder my either disregard the affected areas as it continues to encode the video sequence, or it may reconstruct the affected areas and use them in the encoding process. Reconstruction is performed by replacing the damaged area by its corresponding area in the previous picture. This

technique however introduces unacceptable delays, as the encoder will have to wait for the decoder to provide the location of damaged areas.

Since video sequences that have been compressed according to the current video compression standards [24, 35, 36, 37] consist of DCT coefficients and motion vectors, one approach to packet loss concealment is to prioritize data as described in [38, 39, 40, 25, 41, 42, 43, 44]. Encoded video data is segmented into low priority data such as high frequency DCT coefficients, and high priority data such as addresses of blocks , motion vectors, and low frequency DCT coefficients. Such prioritization is performed since motion vectors are crucial to the reconstruction of motion predicted regions. Likewise, most of the information for reconstructing blocks of data are available in the low frequency DCT coefficients. In the case of ATM networks, prioritization is achieved by setting the cell loss priority (CLP) priority bit of every cell carrying information that is deemed to be highly crucial to the proper reconstruction of the decompressed video sequence to zero [1]. In the event of network congestion, packets carrying low priority data are discarded while those carrying high priority data are retained.

In [42] the transmission of spatially scalable MPEG-2 video over ATM networks is discussed. Video is coded at different resolutions, with the lowest resolution being sent separately in high priority cells and the other resolution streams packed into low priority cells. At the decoder the low resolution frames are first retrieved and reconstructed. To attain higher resolutions, the decoded low resolution frames are then subsequently interpolated.

An additional technique to reduce the effect of packet loss was proposed in [25]. It is based on the fact that the bit rate of a video sequence can be controlled by the quantization scale q, and by N, the number of intraframes in a sequence. Increasing q results in a decrease in the bit rate. Similarly, decreasing the number of intracoded frames will decrease the bit rate. If the packet loss rate temporarily increases, a

[1]The cell loss priority flag is a single bit present in the 5 byte header of each cell. When set to 0, this bit indicates that its cell has higher priority than cells with CLP bits set to 1 [10]

counter measure would then be to decrease N and increase q. This results in a smaller bit rate and hence a smaller packet loss rate.

Alternative techniques to packetization that rely on interleaving of data are proposed in [41, 45, 46, 47]. The underlying idea behind either schemes in [45, 46] is to pack data, such as adjacent blocks or DCT coefficients from the same block into different ATM cells to avoid a complete loss of whole blocks in the event that cell loss occurs. In [41] interleaving is performed at the slice level. It was found that packing every other slice into consecutive cells was sufficient to localize the effect of cell loss, at the mean cell loss rate of 10^{-3}, to a slice, and its neighboring slices kept intact. In [47], each frame is split into four bands and the data for each band packed together. This distributes the error due to cell loss over the entire frame, rather than localize it.

Post processing techniques for the sake of error concealment are addressed in [48, 49, 46, 50, 51, 52, 53, 54, 55, 56, 57]. In [48, 49, 46] it is assumed that partial loss of DCT coefficients has occurred. High frequency coefficients that have been lost are replaced by zeros. The remaining coefficients are estimated, by minimizing a cost function. The cost function is the sum, over all pixels in a block, of the square of the differences between adjacent coefficients. The difference between each coefficient and its neighbors is weighted by either zero or one, depending on whether or not that difference is to be included in the cost function. The method proposed in [50] was proposed for JPEG images but can be applied to MPEG sequences as well. To perform error concealment, histograms of the DCT coefficients at locations (0,0), (0,1), (1,0), and (1,1) in intact regions, are compiled. These are then analyzed for correlations. The coefficients belonging to the missing macroblock are then estimated from those of the neighboring macroblocks. Instead of using all neighboring coefficients, only those that are believed to be correlated, based upon the histograms, are used in the reconstruction process. Using quadratic and linear polynomials to interpolate missing pixel values have also been investigated in [50]. An approach that estimates missing edges in each block from edges in the surrounding blocks was proposed in [52].

For each direction of an estimated edge, a version of the lost block is reconstructed by performing a number of one dimensional interpolations carried out along several lines parallel to the edge. A final version of the missing block is obtained by merging all previously attained versions together. In [51], lost macroblocks are constructed by temporal replacement or spatial interpolation. The decision to use a particular method is based on a measure of image activity. If there is no predominant motion within a frame then temporal replacement is used, otherwise the macroblock is reconstructed by means of spatial interpolation. The method of projections onto convex sets [58] for error concealment is described in [53]. To reconstruct a missing macroblock, it is first determined whether an edge actually a passes through it. This is achieved by applying the Sobel operator [23] to the surrounding macroblocks. A larger block of pixels is then constructed from initial pixel values of the lost macroblock and those of its neighbors. The Discrete Fourier Transform of the larger block is then obtained, and all coefficients lying outside a certain band that is oriented perpendicular to the edge direction are set to zero. The inverse transform is then obtained and those pixel values belonging to intact macroblocks are reset to their original values. The process is then repeated. Since the constraints imposed are convex sets, the process is guaranteed to converge.

In [41] a scheme that utilizes a hybrid of temporal and spatial interpolation is described. Again the decision to use spatial, or temporal, or both temporal and spatial interpolation is based on the amount of motion within a frame.

In all of the above techniques there has been no mention of how to pack Transport Streams such that the loss of macroblocks is detected. Furthermore, there been no reference as to how bits pertinent to one macroblock are distinguished from those belonging to another macroblock in the event that packet loss has occurred. Our approach allows for the detection of macroblock loss and the recognition of the location of the lost macroblocks. The decoder then does not have to decide whether a group of bits belongs to a damaged macroblock or to an undamaged one. This reduces the complexity of the decoding process.

The above mentioned error concealment techniques are known as *passive* error concealment techniques whereby the video stream is post-processed to reconstruct the missing data. An alternative group of techniques known as *active* error concealment techniques also exist.

In active concealment, error control coding techniques are employed to render the coded images/video resilient to channel errors. This entails the addition of extra data for protection purposes. The addition of extra bits makes it necessary to reduce the source coder's data rate to avoid increasing network congestion. Active concealment has the advantage of permitting perfect reconstruction at the decoding end, if the amount of data lost is within the limit that the error control coding scheme can withstand [59, 60].In addition unequal error protection can be provided by varying the number of bits used according to the priority of the data being protected [2, 3, 61, 62, 63, 64, 65]. Although passive concealment does not result in perfect reconstruction of the lost data, it is necessary in many applications where error control coding cannot be used due to compliance with video transmission standards or when active concealment fails. A general overview with an extensive bibliography of the various error concealment methods can be found in [4].

5. ERROR CONCEALMENT IN THE IMAGE AND VIDEO COMPRESSION STANDARDS

The MPEG and H. 263+ video compression standards [24, 35, 37, 66] do not propose any error concealment strategies. However, certain measures were proposed that render MPEG-2, H. 263, and MPEG-4 video streams more error resilient [35, 67, 68, 69].

In MPEG-2, if the decoder detects erroneous bits, it parses the bit stream until the next start code [35]. Thus, increasing the number of slices and hence, the number of slice start codes within a frame will limit the extent of the damage. The affected regions within an image are then reconstructed via motion compensated and/or spatial error concealment. Motion compensated error concealment is done either by using temporal replacement (a zero motion vector), or by averaging the motion vectors belonging to the macroblocks above and below a missing macroblock [35]. Spatial error concealment is achieved by interpolating the missing DC and lowest frequency AC coefficients from those belonging to the surrounding macroblocks [35]. The interpolation scheme, however, is unspecified. The resilience of the video stream can be further enhanced by using any of the Spatial Scalability, Temporal Scalability, Data Partitioning profiles and sending the base layer over a "lossless" channel [35].

Recently the periodic insertion of Resynchronization Markers (unique codes that limit the effect of data loss and establish synchronization between encoder and decoder) has been considered for the MPEG-4 and JPEG2000 standards [67, 68]. The resilience of MPEG-4 bitstreams is also increased by placing coded motion vector data prior to coded DCT coefficients. These two groups of data are separated by a field known as the Motion Boundary Marker (MBM) [67, 68]. The MBM is utilized by the

decoder to discern whether coded motion vectors have been corrupted or not. Uncorrupted coded motion vectors are used to reconstruct macroblocks. If however the motion vector data between two resynchronization markers has been corrupted, all the data between the two markers is discarded and the macroblocks constructed via temporal replacement from the previous frame. The robustness of the stream is further enhanced by using reversible variable length codes that can be uniquely forward or backward decoded, as well as by repeating important header information [67, 68].

An alternative set of strategies have been proposed for making H. 263+ video streams more robust. These include using a 19 bit Bose-Chaudhuri-Hocquenghem (BCH) [70, 71] forward error correction code on every 492 bits, and rectangular slices that can be sent in random order [69]. In addition, if a particular image segment remains unchanged between two I frames, it can be decoded independently. This serves to eliminate error propagation, since if that region were corrupted, it could be restored by temporal replacement from one of the I frames. Damaged frames can also be restored by predicting them from earlier undamaged reference frames.

6. CELL PACKING

In this chapter we will discuss the issues that arise when packing MPEG sequences into network packets. The network protocol that we will be using is the asynchronous transfer mode (ATM) protocol, however the problems faced and the packing procedure set forth apply in general to packet switched networks.

As previously mentioned, an MPEG video stream consists of a sequence header, Group of Pictures (GOP) layer data, picture layer data, slice layer data, and macroblock data. Each layer, except the macroblock data, starts with a 32 bit header. Macroblocks are addressed relative to each other except for the first macroblock in a slice. This is addressed relative to the slice to which it belongs, and relative to the last macroblock in the previous row.

In an MPEG-1 video sequence, consisting mainly of P and B frames, the sizes of the macroblocks will be smaller than the size of the user payload in an ATM cell. This is a consequence of the fact that coding most macroblocks in P or B pictures requires the coding of motion vectors and prediction errors, rather than the DCT coefficients of actual pixel values. This is corroborated by Figures 6.1, 6.2, 6.3, 6.4, and 6.5 respectively, which consist of the histograms of the sizes, in bits, of the macroblocks belonging to the *flowergarden*, *football*, *hockey*, *salesman*, and *table tennis* sequences respectively, when compressed using an MPEG-1 encoder. The histogram bins are each 1 bit in size.

The uncompressed *flowergarden*, *football*, *hockey*, and *table tennis* sequences consist of 150, 720×480 4:1:1 images, whereas the uncompressed *salesman* sequence consists of 300, 4:1:1 CIF images.

The encoder parameters were set such that, each GOP consisted of the following pattern of pictures: IBBPBBPBBPBBPBB, with the quantization scales for the I, P, and B pictures being 8, 10, and 25, respectively. Motion vectors were determined by implementing the logarithmic search strategy, using half pixel displacements over a 10×10 search range. For bidirectionally predicted pictures, the best forward and backward motion vectors were found. The forward motion vector was retained, and a search was carried out for the backward motion vector, that with the available forward motion vector resulted in the best match. The search can alternatively be performed by fixing the backward motion vector, and searching for the forward motion vector. The resulting bit rates for *flowergarden*, *football*, *hockey*, *table tennis*, and *salesman* were 5.05 Mbits/s, 3.81 Mbits/s, 2.53 Mbits/s, 2.74 Mbits/s, and 309.5 Kbits/s respectively.

In the event that cells are lost, the relative addressing of macroblocks is destroyed. Furthermore, information pertinent to the motion vectors, DCT coefficients, quantization scale and type of block is also destroyed. In this case, when attempting to reconstruct the macroblock, the decoder will use data belonging to other macroblocks.

In order to minimize the effect of cell loss, the transmitting ATM Adaptation layer (AAL) could try to segment the incoming data in such a way that no macroblock data is split across cell boundaries. This would require that the AAL know the structure of the incoming data. This can be implemented at the convergence sublayer. The AAL's task then would be to compare the remaining space in the protocol data unit (PDU) with the size of an incoming macroblock. If the macroblock is greater in size than the remaining space, it is placed in a consecutive cell and the remaining space in the cell padded with zeros. Such a technique would be very efficient if the sizes of all the macroblocks in the sequence were much smaller than 48 bytes, the user payload in an ATM cell. Figure 6.6 illustrates this point. It depicts the increase in data rate when an artificial sequence consisting entirely of 7 bit macroblocks is packed into ATM cells. The percentage increase in the data rate is displayed as a function of the size of the ATM cells used. The sizes of the ATM cells were varied such that their user

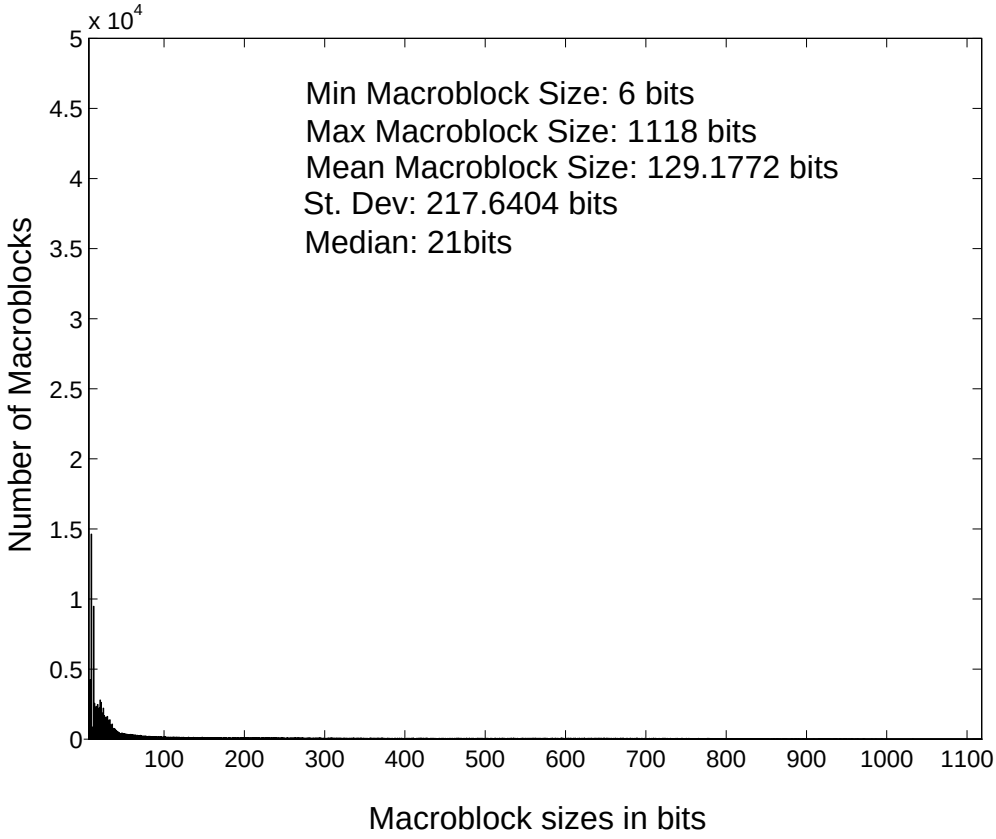


Fig. 6.1. Histogram of the sizes, in bits, of the macroblocks belonging to the compressed version of the *flowergarden* sequence. The bin sizes are 1 bit large. The horizontal axis represents the available sizes of the macroblock, whereas the vertical axis represents the number of macroblocks, within the sequence, that are of the size indicated by each bin.

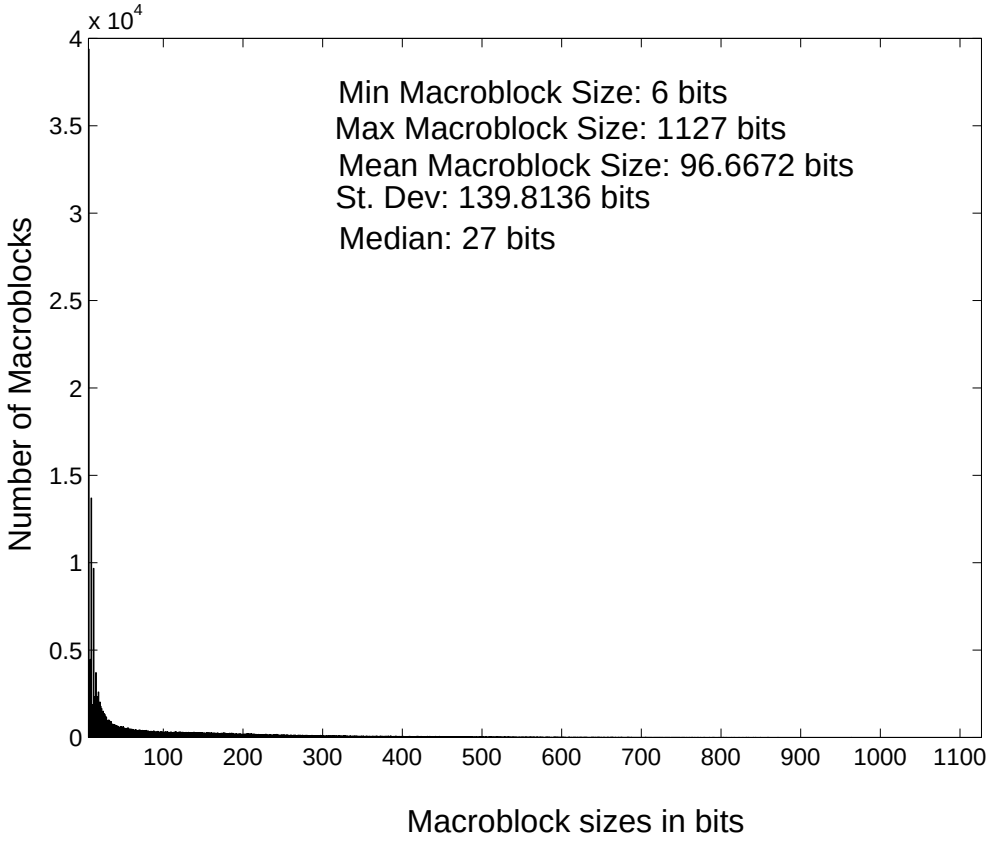


Fig. 6.2. Histogram of the sizes, in bits, of the macroblocks belonging to the compressed version of the *football* sequence. The bin sizes are 1 bit large. The horizontal axis represents the available sizes of the macroblock, whereas the vertical axis represents the number of macroblocks, within the sequence, that are of the size indicated by each bin.

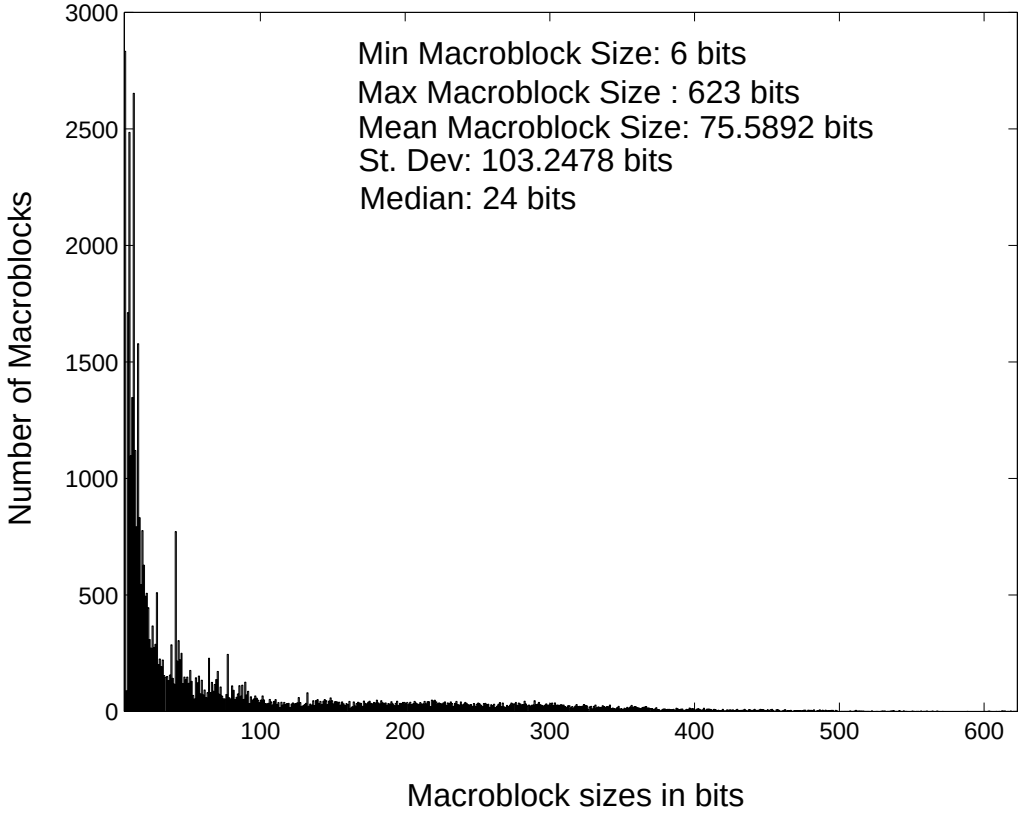


Fig. 6.3. Histogram of the sizes, in bits, of the macroblocks belonging to the compressed version of the *hockey* sequence. The bin sizes are 1 bit large. The horizontal axis represents the available sizes of the macroblock, whereas the vertical axis represents the number of macroblocks, within the sequence, that are of the size indicated by each bin.

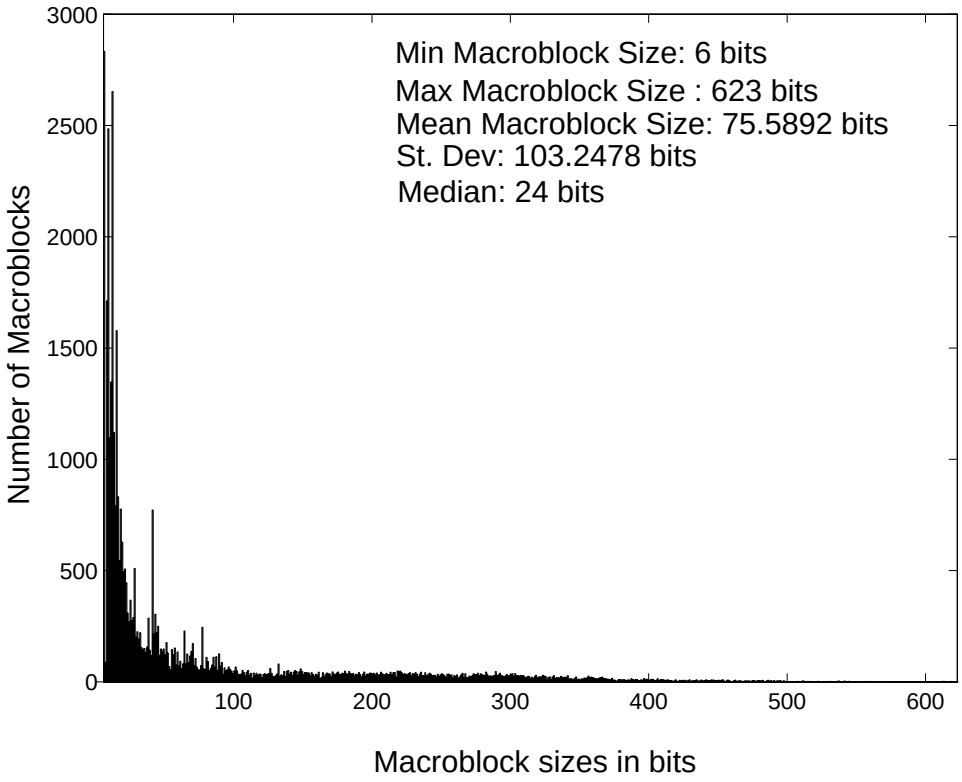


Fig. 6.4. Histogram of the sizes, in bits, of the macroblocks belonging to the compressed version of the *salesman* sequence. The bin sizes are 1 bit large. The horizontal axis represents the available sizes of the macroblock, whereas the vertical axis represents the number of macroblocks, within the sequence, that are of the size indicated by each bin.

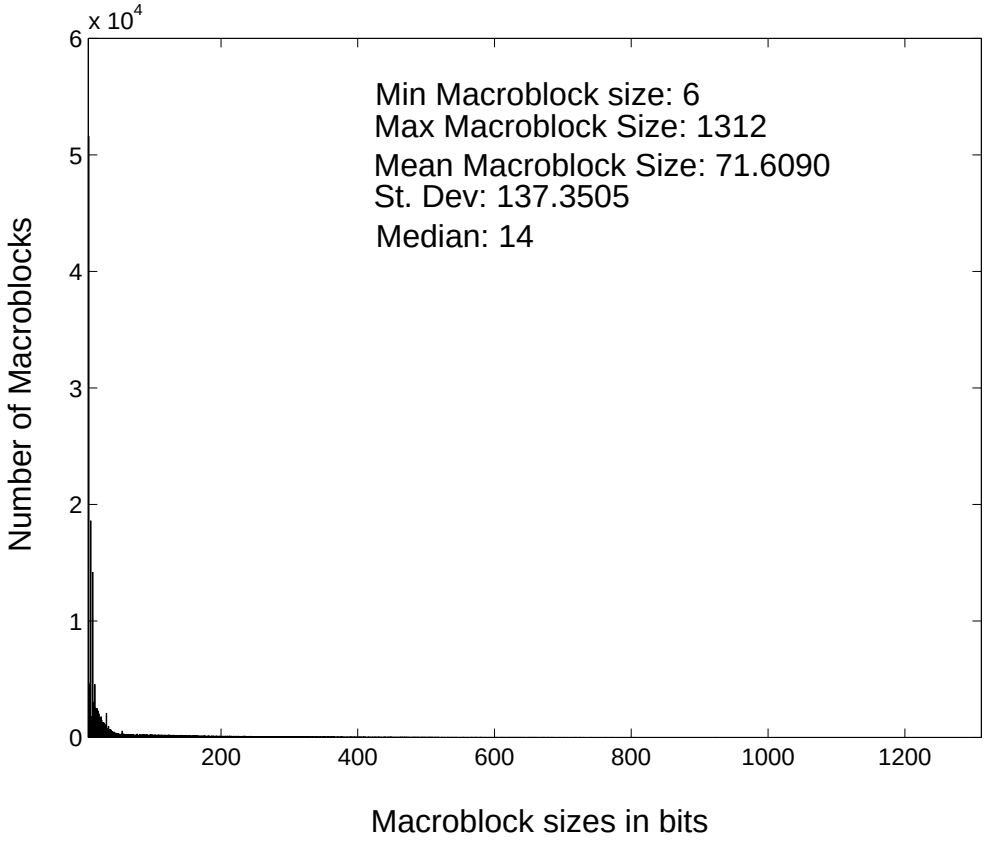


Fig. 6.5. Histogram of the sizes, in bits, of the macroblocks belonging to the compressed version of the *table tennis* sequence. The bin sizes are 1 bit large. The horizontal axis represents the available sizes of the macroblock, whereas the vertical axis represents the number of macroblocks, within the sequence, that are of the size indicated by each bin.

payload sizes were multiples of 48 bytes, and the corresponding percentage increase in the data rate measured. It is evident that the maximum increase in the data rate is 1.6% when the ATM cells consisted of the 48 bytes of user payload. The upper bound shown, is the percentage increase in data rate when the extra zeros appended at the end of each cell are one bit less than the size of the macroblock.

Actual MPEG sequences contain macroblocks of various sizes, including those whose sizes are greater than the size of an ATM cell user payload. To investigate the effect of packing such macroblocks, we repeated the packing procedure for another artificial sequence containing macroblocks that were 400 bits in size. The resulting increase in data rate, corresponding to various ATM cell sizes used, is illustrated in Figure 6.7. In this case, the increase in data rate fell below 10% when the user payload size was greater than 480 bytes. Including 7 bit macroblocks in the sequence offered no improvement. Two more artificial sequences containing alternating 7 and 400 bit macroblocks ware packed into ATM cells. One sequence consisted of 7 bit macroblocks followed by 400 bit macroblocks, whereas the other consisted of 400 bit macroblocks followed by 7 bit macroblocks. The resulting percentage increase in the data rates, for various ATM cell sizes, are depicted in 6.8 and Figures 6.9 respectively. In view of the above, and the fact that MPEG sequences have variable length macroblocks, such a packing scheme would be extremely inefficient. This is corroborated by Figures 6.10, 6.11, 6.12, 6.13, and 6.14 respectively, which illustrate the ensuing increases in data rates when such a packing scheme is employed to pack the compressed *flowergarden*, *football*, *hockey*, *salesman*, and *table tennis* sequences respectively, into ATM cells with different user payload sizes.

To avoid such excessive increase in data rates we present a new packing scheme. Our approach to packing cells with a user payload of 48 bytes is to pack GOP, picture and slice data into high priority cells. This guarantees the safe arrival of synchronizing information present in the GOP, picture and slice layers. All other data is packed into low priority cells. An extra 9 bits, as shown in Figure 6.15, is inserted at the start of each cell to indicate the location of the first macroblock being packed into

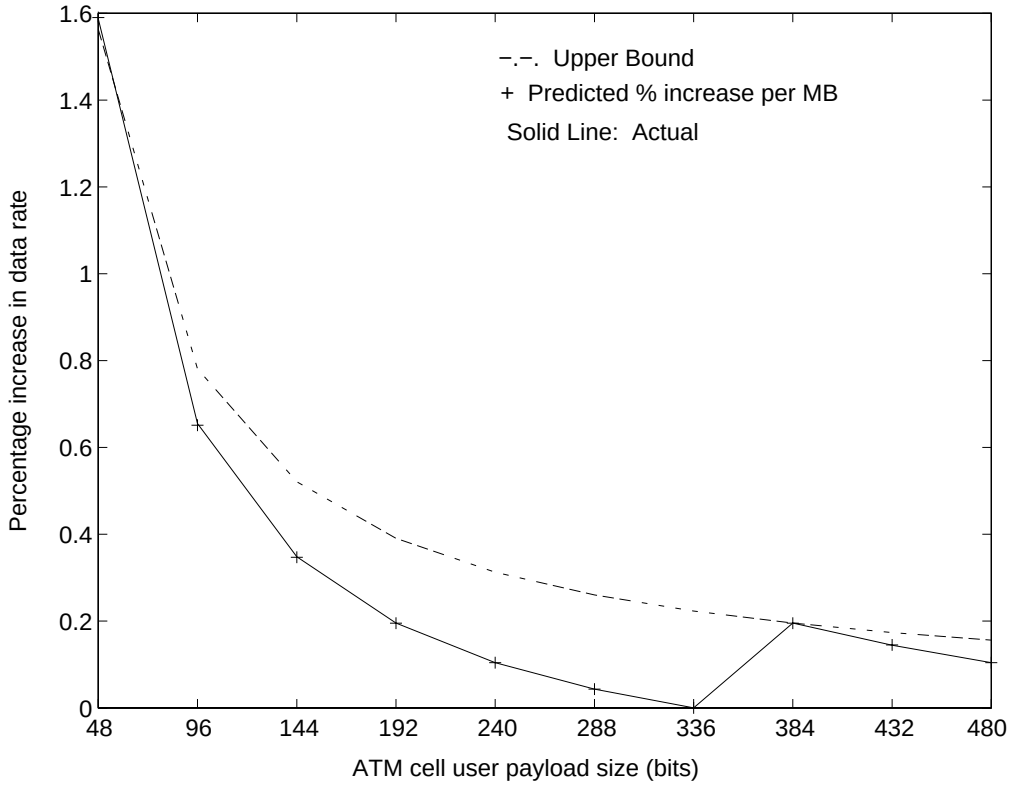


Fig. 6.6. Percentage increase in the data rate when a sequence containing 7 bit macroblocks is packed into ATM cells. The macroblocks are packed in such a way that no macroblock is split between two cells. If the size of an incoming macroblock is greater than the remaining space in a cell, it would then be placed in the next cell and the remaining space filled with zeros. The horizontal axis represents the size of the user payload, as a multiple of 48 bytes, and the vertical axis depicts the percentage increase in data rate corresponding to the cell size used.

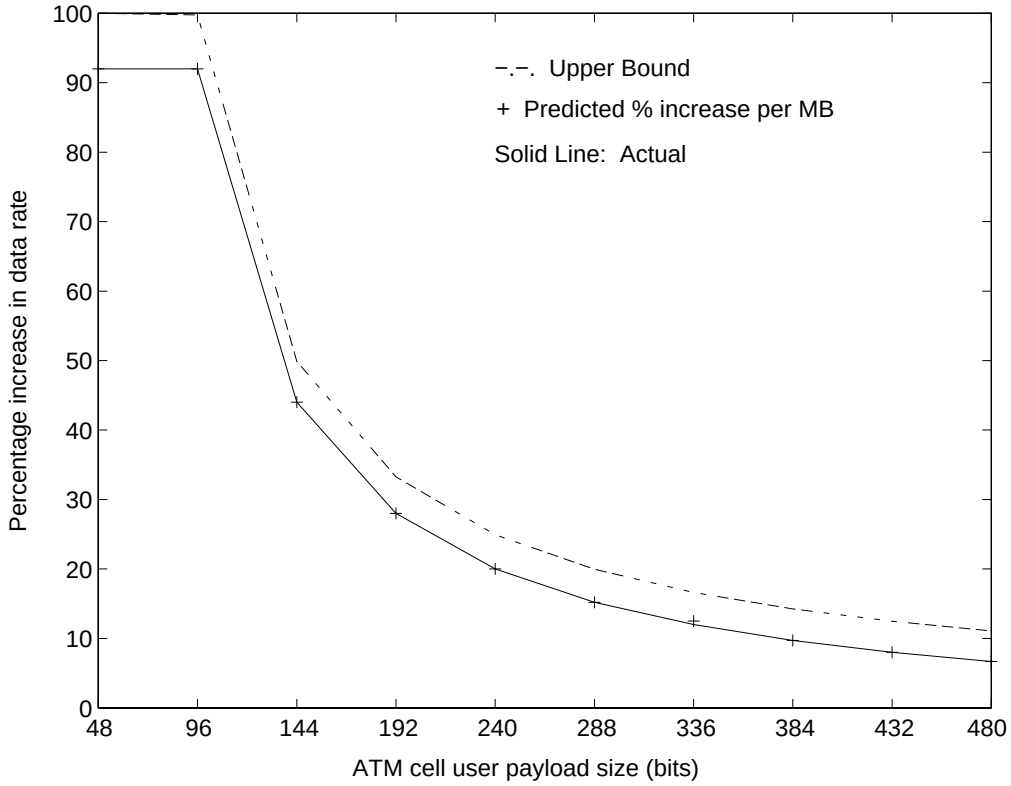


Fig. 6.7. Percentage increase in data rate when a sequence containing 400 bit macroblocks is packed into ATM cells. The macroblocks are packed in such a way that no macroblock is split between two cells. If the size of an incoming macroblock is greater than the remaining space in a cell, it would then be placed in the next cell and the remaining space filled with zeros. The horizontal axis represents the size of the user payload, as a multiple of 48 bytes, and the vertical axis depicts the percentage increase in data rate corresponding to the cell size used.

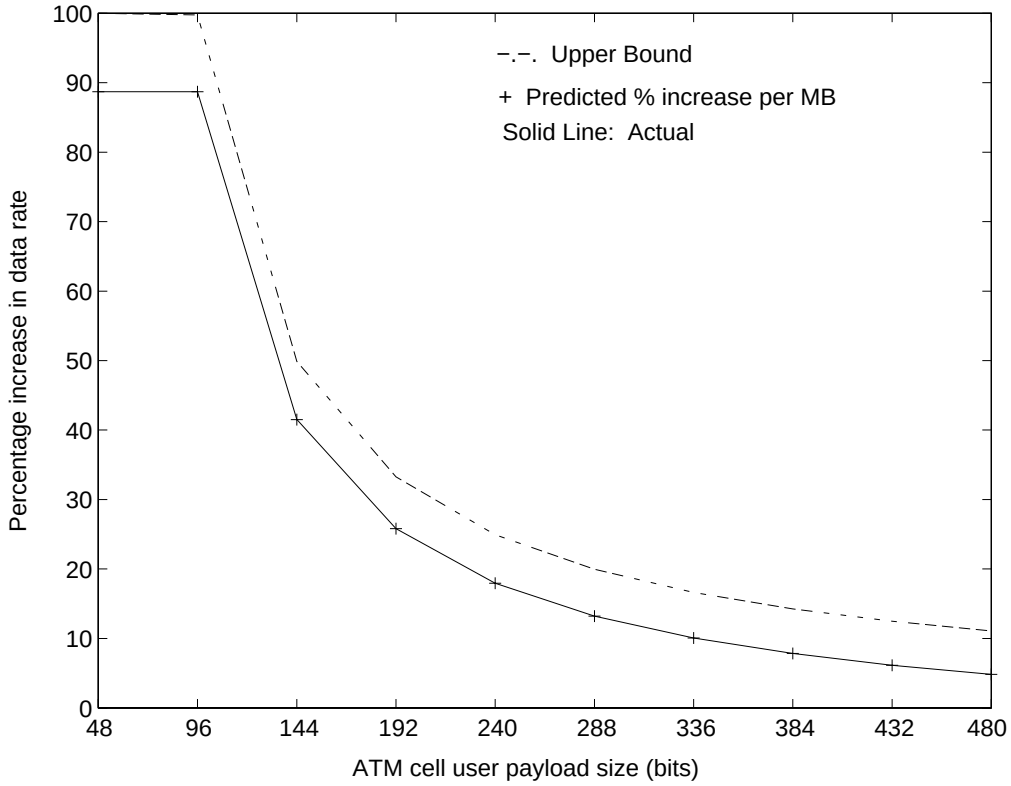


Fig. 6.8. Percentage increase in data rate when a sequence consisting of alternating 7 and 400 bit macroblocks is packed into ATM cells. The macroblocks are packed in such a way that no macroblock is split between two cells. If the size of an incoming macroblock is greater than the remaining space in a cell, it would then be placed in the next cell and the remaining space filled with zeros. The horizontal axis represents the size of the user payload, as a multiple of 48 bytes, and the vertical axis depicts the percentage increase in data rate corresponding to the cell size used.

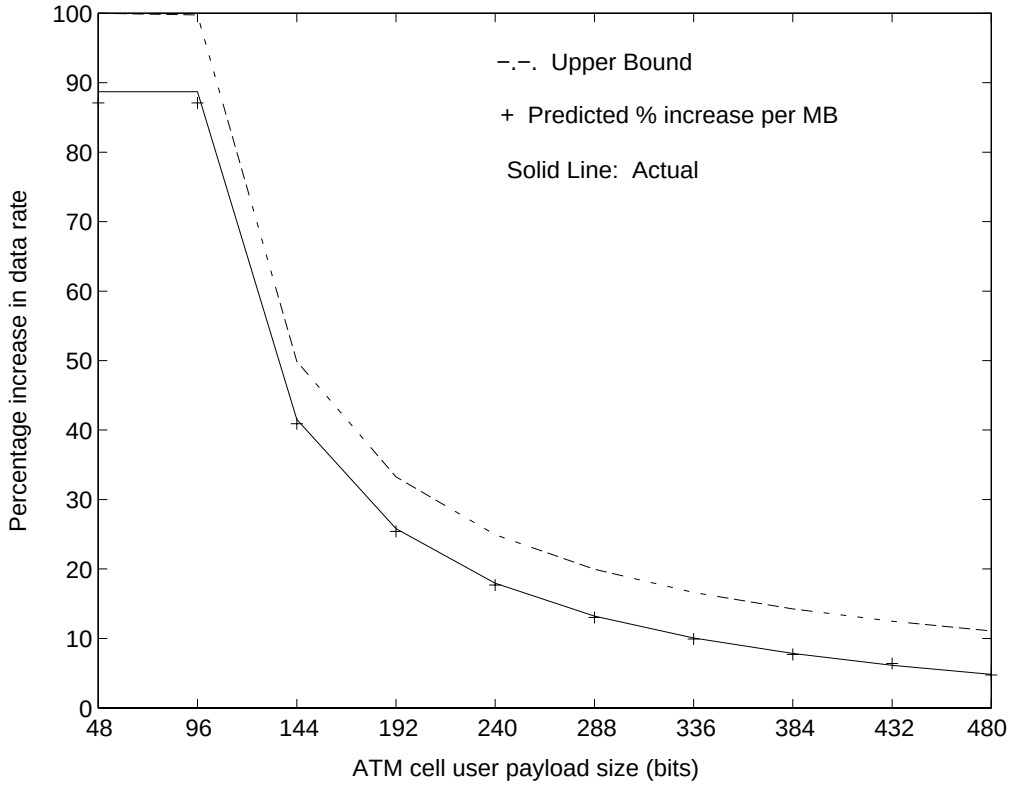


Fig. 6.9. Percentage increase in data rate when a sequence consisting of alternating 400 and 7 bit macroblocks is packed into ATM cells. The macroblocks are packed in such a way that no macroblock is split between two cells. If the size of an incoming macroblock is greater than the remaining space in a cell, it would then be placed in the next cell and the remaining space filled with zeros. The horizontal axis represents the size of the user payload, as a multiple of 48 bytes, and the vertical axis depicts the percentage increase in data rate corresponding to the cell size used.

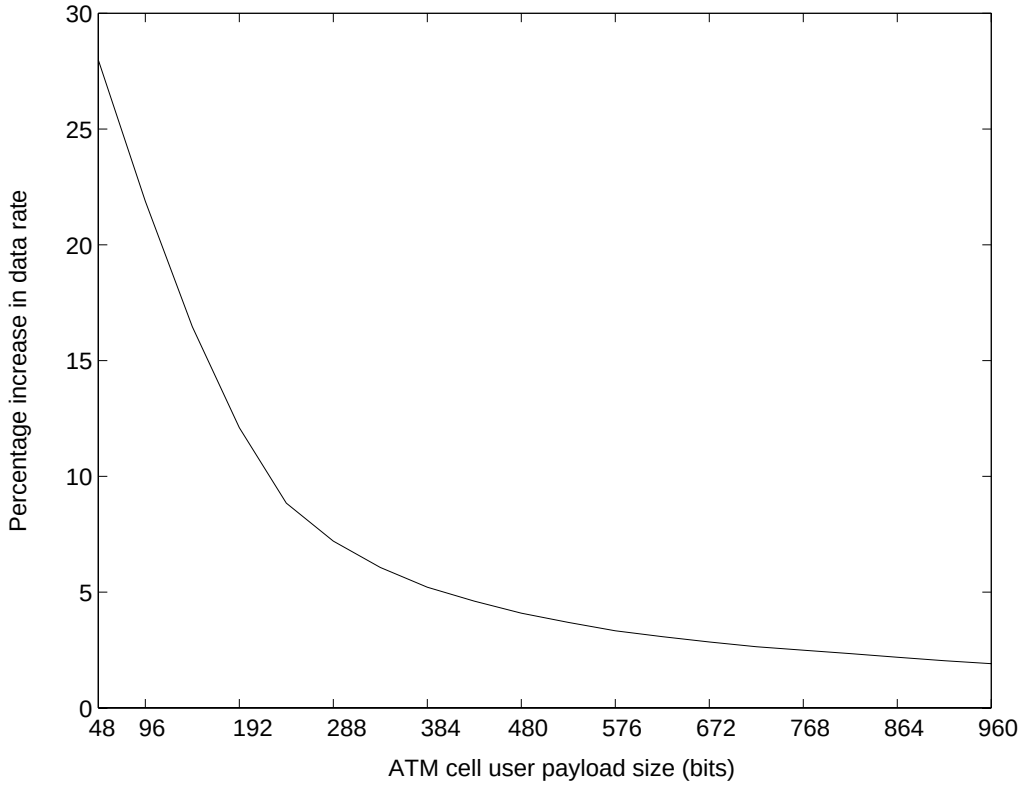


Fig. 6.10. Percentage increase in data rate when the compressed *flowergarden* sequence is packed into ATM cells. The macroblocks are packed in such a way that no macroblock is split between two cells. If the size of an incoming macroblock is greater than the remaining space in a cell, it would then be placed in the next cell and the remaining space filled with zeros. The horizontal axis represents the size of the user payload, as a multiple of 48 bytes, and the vertical axis depicts the percentage increase in data rate corresponding to the cell size used.

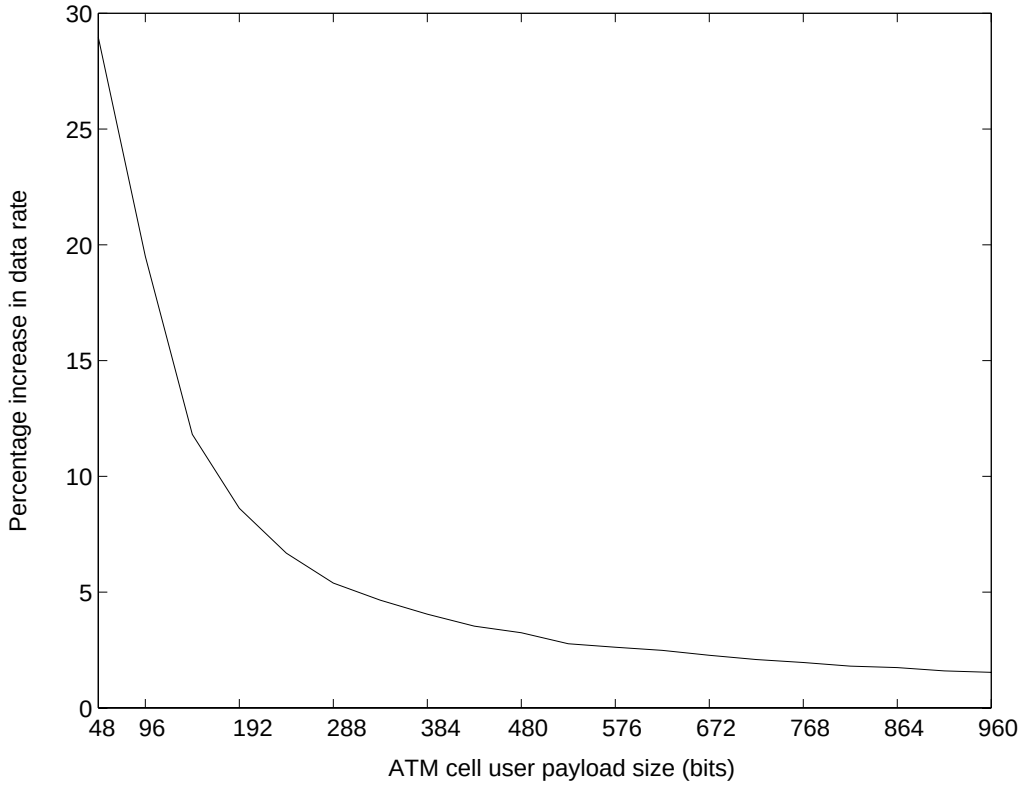


Fig. 6.11. Percentage increase in data rate when the compressed *football* sequence is packed into ATM cells. The macroblocks are packed in such a way that no macroblock is split between two cells. If the size of an incoming macroblock is greater than the remaining space in a cell, it would then be placed in the next cell and the remaining space filled with zeros. The horizontal axis represents the size of the user payload, as a multiple of 48 bytes, and the vertical axis depicts the percentage increase in data rate corresponding to the cell size used.

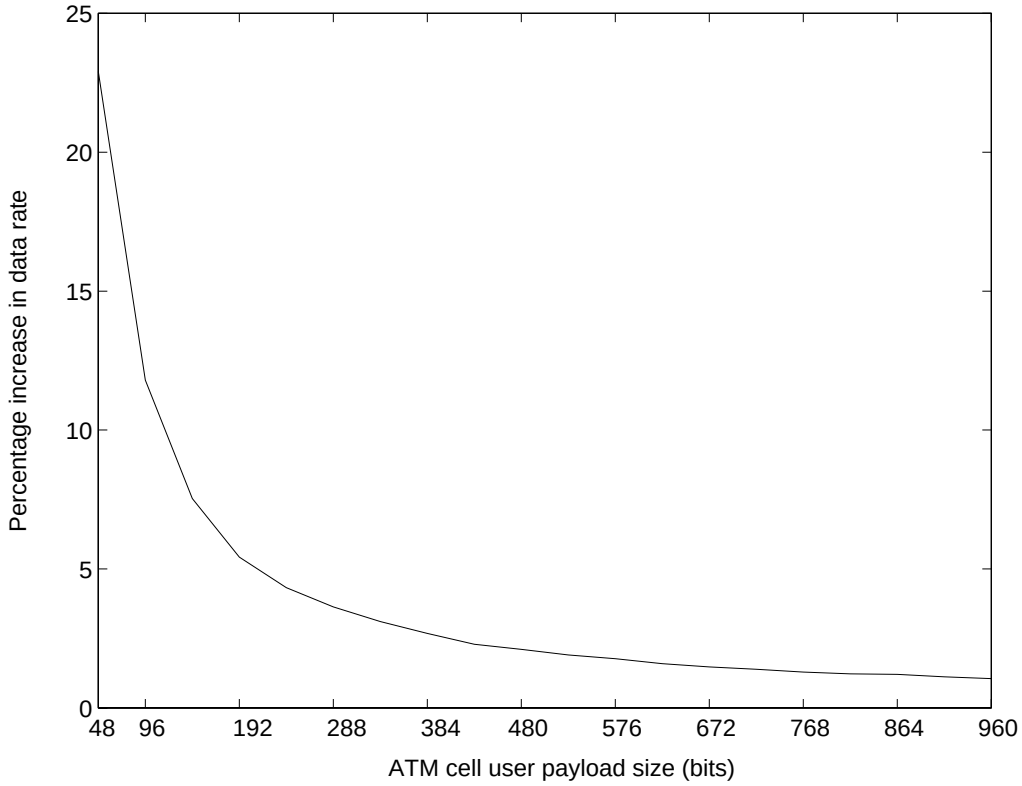


Fig. 6.12. Percentage increase in data rate when the compressed *hockey* sequence is packed into ATM cells. The macroblocks are packed in such a way that no macroblock is split between two cells. If the size of an incoming macroblock is greater than the remaining space in a cell, it would then be placed in the next cell and the remaining space filled with zeros. The horizontal axis represents the size of the user payload, as a multiple of 48 bytes, and the vertical axis depicts the percentage increase in data rate corresponding to the cell size used.

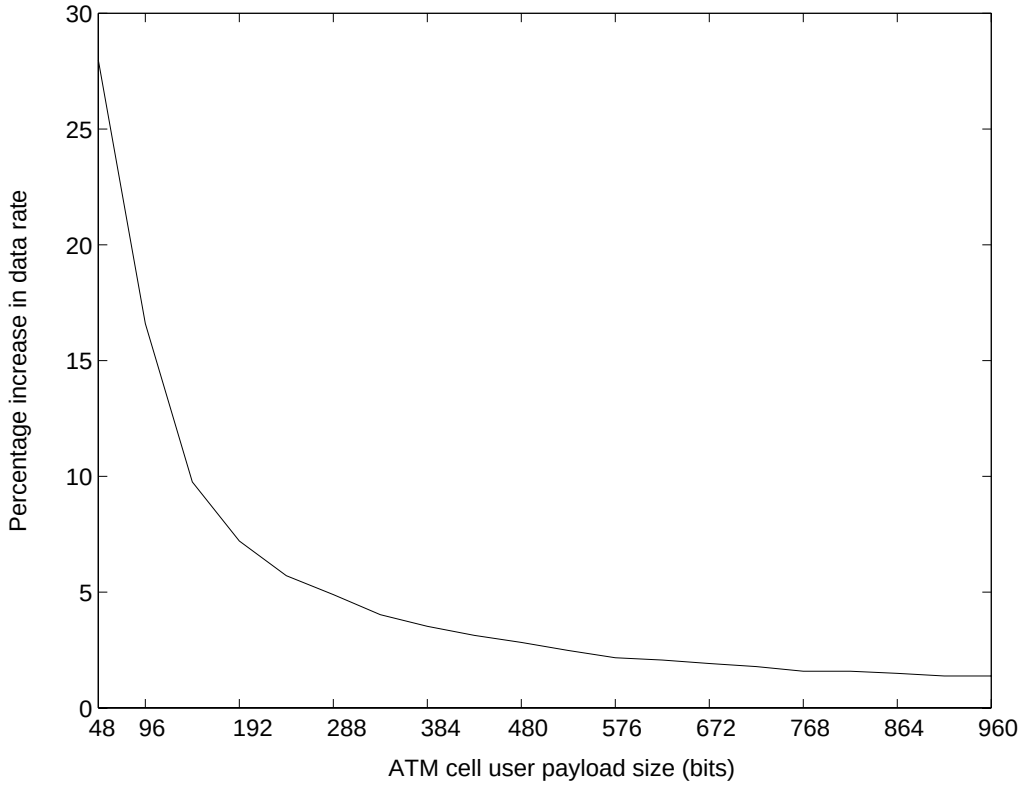


Fig. 6.13. Percentage increase in data rate when the compressed *salesman* sequence is packed into ATM cells. The macroblocks are packed in such a way that no macroblock is split between two cells. If the size of an incoming macroblock is greater than the remaining space in a cell, it would then be placed in the next cell and the remaining space filled with zeros. The horizontal axis represents the size of the user payload, as a multiple of 48 bytes, and the vertical axis depicts the percentage increase in data rate corresponding to the cell size used.

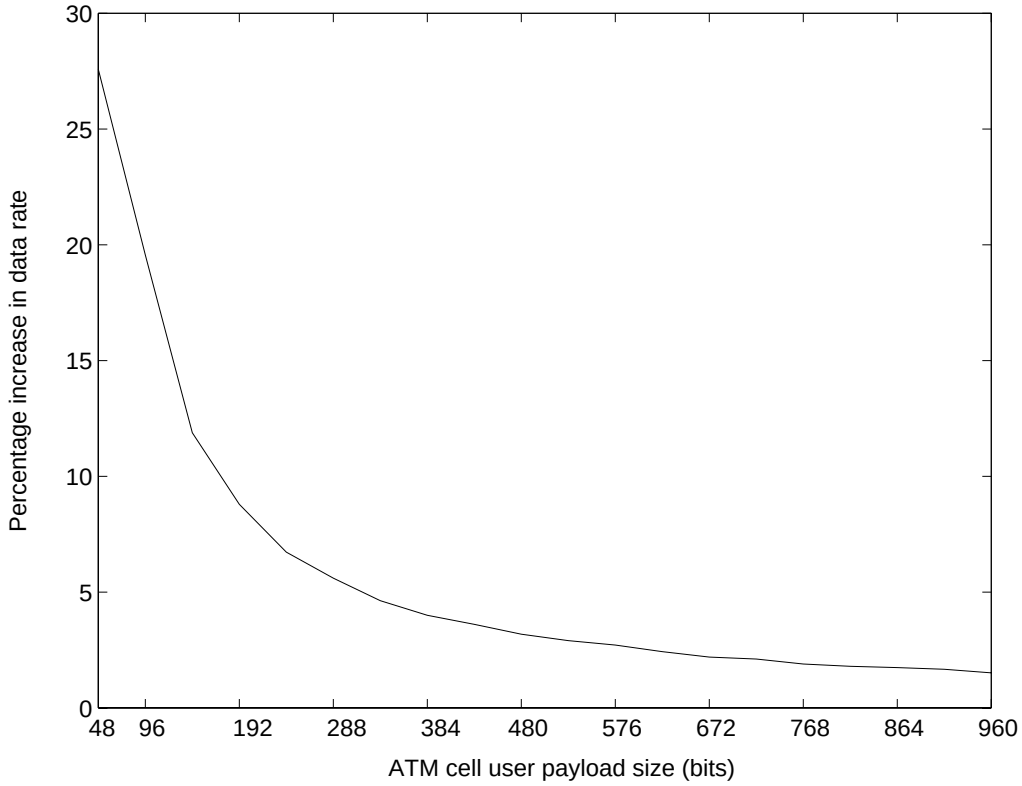


Fig. 6.14. Percentage increase in data rate when the compressed *table tennis* sequence is packed into ATM cells. The macroblocks are packed in such a way that no macroblock is split between two cells. If the size of an incoming macroblock is greater than the remaining space in a cell, it would then be placed in the next cell and the remaining space filled with zeros. The horizontal axis represents the size of the user payload, as a multiple of 48 bytes, and the vertical axis depicts the percentage increase in data rate corresponding to the cell size used.

the cell. These extra bits are also used to indicate when a macroblock spans across more than one cell. They are then followed by another 7 bits that are used to provide the relative address of the macroblock with respect to the slice in which it is located. The extra 16 bits are used to localize the loss of macroblocks within a frame while maintaining the ease of decoding the correctly received macroblocks. This is crucial to MPEG streams, since a loss of a few bits can perturb the decoding process and result in the loss of entire frames. At the receiving end the 16 bits are stripped off and the MPEG decoder is provided with an MPEG stream that has missing macroblocks. The addressing of all intact macroblocks remains unaltered. Several sequences were packed according to the new packing scheme. The overall increase in the data rate of every sequence was measured and found to be less than 7% of the original data rate [54]. It is to be noted that 16 is the minimum number of bits necessary to provide the required information when CCIR601 images are coded such that each slice consists of an entire row of macroblocks.

This packing scheme can be applied to other packet switched networks. The only difference being that the extra bits inserted in the user payload portion will have to be adjusted to fit with the size of the packet's user payload section.

The packing scheme described here was applied to MPEG-1 video streams only. In a real world situation however, video streams are multiplexed with audio streams and are delivered as Transport Streams. Thus, the packing of Transport Streams into ATM cells in such a way to facilitate the identification and recovery of damaged video data needs to be addressed.

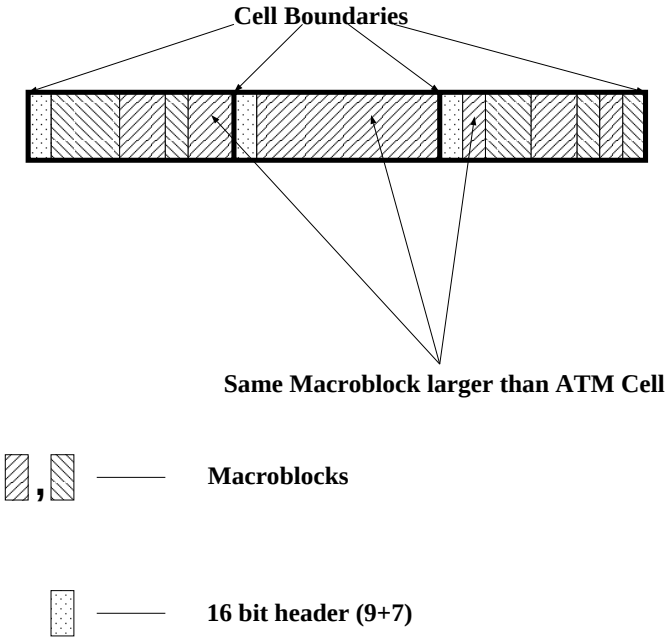


Fig. 6.15. Packing scheme. Extra 9 bits are inserted at the start of each cell to indicate the location of the first macroblock being packed into the cell. These extra bits are also used to indicate when a macroblock spans across more than one cell. Another 7 bits are appended to the 9 bits, and used to provide the relative address of the macroblock with respect to the slice in which it is located. These extra 16 bits are used to localize the loss of macroblocks within a frame while maintaining the ease of decoding the correctly received macroblocks.

7. ERROR CONCEALMENT

7.1 Introduction

The goal of passive error concealment is to estimate missing data. In the case of MPEG video, the objective is to estimate missing macroblocks and motion vectors. The underlying idea is that there is still enough redundancy in the sequence to be exploited by the concealment technique. In particular, in I frames it is possible to have a lost macroblock surrounded by intact macroblocks that are used to interpolate the missing data. This is a result of the fact that macroblocks in I frames can span across two packets. It also arises when the macroblocks in I frames are interleaved prior to packing them into packets. In P and B pictures, it is possible to have entire rows of macroblocks missing. In this case, spatial interpolation will not yield acceptable reconstructions. However, the motion vectors of the surrounding regions can be used to estimate the lost vectors, and the damaged region reconstructed via motion compensated interpolation [54, 55, 56].

Let X be an $N_1 \times N_2$ decompressed frame from an MPEG sequence and let Y be the received version at the output of the channel. Due to channel errors it is conceivable that Y may have missing data. Each transmitted picture consists of M macroblocks that have $N \times N$ pixels. Let $\mathbf{x}_i$ be the lexicographic ordering of the pixels in the i^{th} macroblock in X. The vector $\mathbf{x}$ is then defined to be the concatenation of $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M$, that is

$$\mathbf{x} = [\mathbf{x}_1^T \ \mathbf{x}_2^T \ \ldots \ \mathbf{x}_M^T]^T. \tag{7.1}$$

The vector $\mathbf{y}$ is similarly defined for Y. If the j^{th} macroblock is missing due to packet loss then

$$\mathbf{y} = \mathbf{D}\mathbf{x}, \tag{7.2}$$

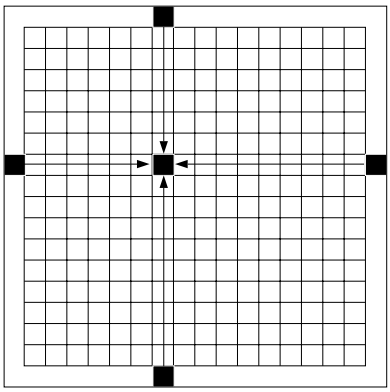


Fig. 7.1. Spatial Averaging. Each lost pixel (shown here in black with arrows pointing to it) is reconstructed from its four closest intact pixels (shown here in black and lying outside the lost macroblock boundary)

where $\mathbf{D}$ is an $(N_1N_2 - N^2) \times N_1N_2$ matrix that consists of the identity matrix excluding the rows from row jN^2 to row $(j+1)N^2 - 1$. If n of the M macroblocks are missing due to packet loss then $\mathbf{D}$ will be an $(N_1N_2 - nN^2) \times N_1N_2$ matrix.

Problem statement: Given the received data $\mathbf{y}$ estimate $\mathbf{x}$.

7.2 Deterministic Spatial Approach

Suppose that the k^{th} macroblock, $\mathbf{x_k}$ has been lost during the process of transmission. Let $\hat{X}_{i,j}$ denote the reconstructed value of the sample at the i^{th} row and j^{th} column of $\mathbf{x_k}$, and let $\mathbf{J_k}$ denote the set of indices of the pixels belonging to $\mathbf{x_k}$, that is

$$\mathbf{J_k} = \{(i,j) \mid X_{i,j} \, \epsilon \, \mathbf{x_k}\}. \tag{7.3}$$

We will discuss two methods for reconstructing a lost macroblocks from its neighbors.

7.2.1 Interpolation

In this method, every pixel in $\mathbf{x_k}$ is reconstructed by spatially averaging the values of its four closest intact neighbors as shown in Figure 7.1.

$$\hat{X}_{i,j} = \lambda[\mu_1 X_{i,-1} + (1-\mu_1)X_{i,N}] + (1-\lambda)[(1-\mu_2)X_{-1,j} + \mu_2 X_{N,j}], \tag{7.4}$$

where

$X_{i,-1}$ - is the closest element in the macroblock to the left of $\mathbf{x_k}$,

$X_{i,N}$ - is the closest element in the macroblock to the right of $\mathbf{x_k}$,

$X_{-1,j}$ - is the closest element in the macroblock above $\mathbf{x_k}$,

$X_{N,j}$ - is the closest element in the macroblock below $\mathbf{x_k}$.

The weighting coefficients μ_1 and $1 - \mu_1$ are used to weigh the contributions from the pixels on either side of the lost pixel, and μ_2 and $1 - \mu_2$ are used to weigh the contributions from those above and below the lost pixel. The coefficient μ_1 is a function of the distances between the lost pixel and its closest neighbors lying on the same row, while μ_2 depends on the distances between the lost pixel and its closest neighbors in the same column. The contributions from the macroblocks on either side are weighted by λ, and those from the ones above and below are weighted by $1 - \lambda$. The advantage of this method is that it is simple, fast, and results in good reconstruction.

An alternative approach has been proposed to estimate missing edges in each block from edges in the surrounding blocks [52]. For each direction of an estimated edge, a version of the lost block is reconstructed by performing a number of one dimensional interpolations carried out along several lines parallel to the edge. A final version of the missing block is obtained by merging all previously attained versions together.

7.2.2 Optimal Iterative Reconstruction

The second proposed method aims at reconstructing the lost macroblocks by minimizing a cost function. This cost function, f, is the sum of the weighted square differences between each lost pixel value and its neighbors [72, 48], which include pixels from surrounding undamaged blocks shown in the dark region in Figure 7.2. Thus,

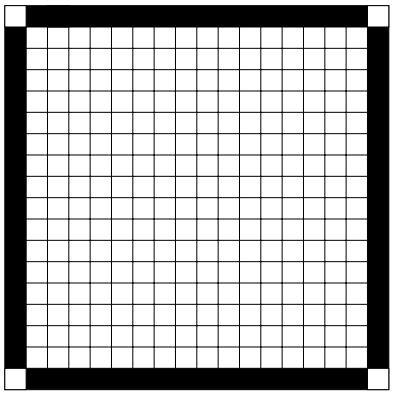


Fig. 7.2. The cost function includes the differences between pixels on the boundary of the lost macroblock and their neighbors. These neighboring pixels that belong to the other macroblocks, are shown here in black.

$$f(\hat{\mathbf{x}}_{\mathbf{k}}, Y) = \frac{1}{2} \sum_{(i,j)\epsilon \mathbf{J}_{\mathbf{k}}} [\omega^w_{i,j}(\hat{X}_{i,j} - \hat{X}_{i,j-1})^2 + \omega^e_{i,j}(\hat{X}_{i,j} - \hat{X}_{i,j+1})^2 + \omega^n_{i,j}(\hat{X}_{i,j} - \hat{X}_{i-1,j})^2 + \omega^s_{i,j}(\hat{X}_{i,j} - \hat{X}_{i+1,j})^2] \quad (7.5)$$

where $\hat{\mathbf{x}}_{\mathbf{k}}$ is the reconstructed version of $\mathbf{x}_{\mathbf{k}}$, and $\omega^w_{i,j}$, $\omega^e_{i,j}$, $\omega^n_{i,j}$ and $\omega^s_{i,j}$ are the weighting coefficients for the pixel values left of, right of, above, and below the current pixel, respectively. The explicit dependence of the cost function on Y is used to indicate that the sum in Equation 7.5 contains pixel values from the dark region shown in Figure 7.2. Since the macroblocks in each frame are numbered in lexicographic ordering then Equation 7.5 can be rewritten as

$$f(\hat{\mathbf{x}}_{\mathbf{k}}, Y) = \frac{1}{2} \sum_{i=\lfloor \mathbf{k}/N \rfloor}^{i=\lfloor \mathbf{k}/N \rfloor + N - 1} \sum_{j=mod(\mathbf{k},N)}^{j=mod(\mathbf{k},N)+N-1} [\omega^w_{i,j}(\hat{X}_{i,j} - \hat{X}_{i,j-1})^2 + \omega^e_{i,j}(\hat{X}_{i,j} - \hat{X}_{i,j+1})^2 + \omega^n_{i,j}(\hat{X}_{i,j} - \hat{X}_{i-1,j})^2 + \omega^s_{i,j}(\hat{X}_{i,j} - \hat{X}_{i+1,j})^2] \quad (7.6)$$

where

$\lfloor a \rfloor$ - is the greatest integer less than or equal to real number a and

$mod(\mathbf{k}, N)$ - is the remainder resulting from having divided $\mathbf{k}$ by N .

The same set of weighting coefficients is used to reconstruct each missing macroblock. Without loss of generality, the indices i and j in Equation 7.6 can be made to vary between 1 and N. Hence

$$f(\hat{\mathbf{x}}_{\mathbf{k}}, Y) = \frac{1}{2} \sum_{i=1}^{i=N} \sum_{j=1}^{j=N} [\omega^w_{i,j}(\hat{X}_{i,j} - \hat{X}_{i,j-1})^2 + \omega^e_{i,j}(\hat{X}_{i,j} - \hat{X}_{i,j+1})^2 + \omega^n_{i,j}(\hat{X}_{i,j} - \hat{X}_{i-1,j})^2 + \omega^s_{i,j}(\hat{X}_{i,j} - \hat{X}_{i+1,j})^2] \quad (7.7)$$

Equation 7.7 can be rewritten as

$$f(\hat{\mathbf{x}}_{\mathbf{k}}, Y) = \frac{1}{2} \hat{\mathbf{x}}^T_{\mathbf{k}} \mathbf{Q} \hat{\mathbf{x}}_{\mathbf{k}} - \hat{\mathbf{x}}^T_{\mathbf{k}} \mathbf{b} + c \quad (7.8)$$

where

$$\mathbf{b} = \sum_{k=1}^{4} [\mathbf{S}_k - \mathbf{Q}_k^T \mathbf{S}_k] \mathbf{b}_k \tag{7.9}$$

$$\mathbf{Q} = \sum_{k=1}^{4} [\mathbf{S}_k - \mathbf{S}_k \mathbf{Q}_k - \mathbf{Q}_k^T \mathbf{S}_k + \mathbf{Q}_k^T \mathbf{S}_k \mathbf{Q}_k] \tag{7.10}$$

$$c = \frac{1}{2} \sum_{k=1}^{4} \mathbf{b}_k^T \mathbf{S}_k \mathbf{b}_k. \tag{7.11}$$

$$\mathbf{S}_1 = \text{diagonal}(\omega_{1,1}^w, \omega_{1,2}^w, \ldots, \omega_{1,N}^w, \omega_{2,1}^w, \omega_{2,2}^w, \ldots, \omega_{2,N}^w, \ldots, \omega_{N,1}^w, \omega_{N,2}^w, \ldots, \omega_{N,N}^w) \tag{7.12}$$

$$\mathbf{S}_2 = \text{diagonal}(\omega_{1,1}^e, \omega_{1,2}^e, \ldots, \omega_{1,N}^e, \omega_{2,1}^e, \omega_{2,2}^e, \ldots, \omega_{2,N}^e, \ldots, \omega_{N,1}^e, \omega_{N,2}^e, \ldots, \omega_{N,N}^e) \tag{7.13}$$

$$\mathbf{S}_3 = \text{diagonal}(\omega_{1,1}^n, \omega_{1,2}^n, \ldots, \omega_{1,N}^n, \omega_{2,1}^n, \omega_{2,2}^n, \ldots, \omega_{2,N}^n, \ldots, \omega_{N,1}^n, \omega_{N,2}^n, \ldots, \omega_{N,N}^n) \tag{7.14}$$

$$\mathbf{S}_4 = \text{diagonal}(\omega_{1,1}^s, \omega_{1,2}^s, \ldots, \omega_{1,N}^s, \omega_{2,1}^s, \omega_{2,2}^s, \ldots, \omega_{2,N}^s, \ldots, \omega_{N,1}^s, \omega_{N,2}^s, \ldots, \omega_{N,N}^s) \tag{7.15}$$

$$\mathbf{b}_1 = [X_{1,0}\ 0\ 0\ \ldots\ 0\ X_{2,0}\ 0\ 0\ \ldots\ 0\ \ldots\ X_{N,0}\ 0\ 0\ \ldots\ 0]^T \tag{7.16}$$

$$\mathbf{b}_2 = [0\ 0\ 0\ \ldots\ X_{1,N+1}\ 0\ 0\ 0\ \ldots\ X_{2,N+1}\ \ldots\ 0\ 0\ 0\ \ldots\ X_{N,N+1}]^T \tag{7.17}$$

$$\mathbf{b}_3 = [X_{0,1}\ X_{0,2}\ X_{0,3}\ \ldots\ X_{0,N}\ 0\ 0\ 0\ \ldots\ 0\ \ldots\ 0\ 0\ 0\ \ldots\ 0]^T \tag{7.18}$$

$$\mathbf{b}_4 = [0\ 0\ 0\ \ldots\ 0\ 0\ 0\ 0\ \ldots\ 0\ \ldots\ X_{N+1,1}\ X_{N+1,2}\ X_{N+1,3}\ \ldots\ X_{N+1,N}]^T \tag{7.19}$$

where diagonal$(\cdot)$ indicates a diagonal matrix with the arguments comprising the diagonal elements of the matrix. The matrices $\mathbf{Q}_k$ are upper and lower diagonal matrices with zeros along the diagonals that satisfy

$$[\mathbf{Q}_1]_{i,j} = \begin{cases} 1 & i = j+1 \text{ and } mod(i,N) \neq 1 \\ 0 & \text{otherwise} \end{cases} \tag{7.20}$$

$$[\mathbf{Q}_3]_{i,j} = \begin{cases} 1 & i = j+N \\ 0 & \text{otherwise} \end{cases} \tag{7.21}$$

$$\mathbf{Q}_2^T = \mathbf{Q}_1 \tag{7.22}$$

$$\mathbf{Q}_4^T = \mathbf{Q}_3 \tag{7.23}$$

.

When all weighting coefficients are greater than zero then, $\mathbf{Q}$ is positive definite, which is necessary for attaining the optimal solution, in the mean square error sense. In this case the optimal solution is

$$\hat{\mathbf{x}}_{\mathbf{k}}^{opt} = \mathbf{Q}^{-1}\mathbf{b}. \tag{7.24}$$

The above equation can be iteratively solved. For instance, the following equations depict the l^{th} iteration for $\mathbf{x}$ in the event that the steepest descent algorithm is used to find $\mathbf{x}$.

$$\mathbf{x}_l = \mathbf{x}_{l-1} - \frac{\mathbf{g}_{l-1}^T\mathbf{g}_{l-1}}{\mathbf{g}_{l-1}^T\mathbf{Q}\mathbf{g}_{l-1}}\mathbf{g}_{l-1} \tag{7.25}$$

where

$$\mathbf{g}_l = \mathbf{Q}\mathbf{x}_l - \mathbf{b} \tag{7.26}$$

is the l^{th} iteration of the gradient vector of f.

To show that $\mathbf{Q}$ is positive definite, we denote the i^{th} diagonal element of $\mathbf{S_1}$ by α_i then

$$\mathbf{S}_1 = \text{diagonal}(\alpha_1, \alpha_2, \ldots, \alpha_{N^2}) \tag{7.27}$$

Using the definition of $\mathbf{S_1}$ and $\mathbf{Q_1}$ we then have

$$[\mathbf{S}_1\mathbf{Q}_1]_{i,j} = \begin{cases} \alpha_i & i = j+1 \text{ and } mod(j,N) \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{7.28}$$

$$[\mathbf{Q}_1^T\mathbf{S}_1]_{i,j} = \begin{cases} \alpha_j & j = i+1 \text{ and } mod(i,N) \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{7.29}$$

$$[\mathbf{Q}_1^T\mathbf{S}_1\mathbf{Q}_1]_{i,j} = \begin{cases} \alpha_{i+1} & j = i \text{ and } mod(i,N) \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{7.30}$$

Thus for any vector $\mathbf{x}$

$$\mathbf{x}^T(\mathbf{S}_1 + \mathbf{Q}_1^T\mathbf{S}_1\mathbf{Q}_1 - \mathbf{Q}_1^T\mathbf{S}_1 - \mathbf{S}_1\mathbf{Q}_1)\mathbf{x} = \sum_{i=1,\ mod(i,N)=1}^{N^2} \alpha_i x_i^2 + \sum_{i=2,\ mod(i,N)\neq 1}^{N^2-1} \alpha_i(x_i - x_{i-1)^2} \tag{7.31}$$

Similarly

$$\mathbf{x}^T(\mathbf{S}_2 + \mathbf{Q}_2^T\mathbf{S}_2\mathbf{Q}_2 - \mathbf{Q}_2^T\mathbf{S}_2 - \mathbf{S}_2\mathbf{Q}_2)\mathbf{x} = \sum_{i=1,\ mod(i,N)=0}^{N^2} \beta_i x_i^2 + \sum_{i=1,\ mod(i,N)\neq 1}^{N^2-1} \beta_i (x_i - x_{i+1)^2} \tag{7.32}$$

$$\mathbf{x}^T(\mathbf{S}_3 + \mathbf{Q}_3^T\mathbf{S}_3\mathbf{Q}_3 - \mathbf{Q}_3^T\mathbf{S}_3 - \mathbf{S}_3\mathbf{Q}_3)\mathbf{x} = \sum_{i=N^2-N+}^{N^2} \gamma_i x_i^2 + \sum_{i=1}^{N^2-N} \gamma_{i+N}(x_i - x_{i+N)^2} \tag{7.33}$$

$$\mathbf{x}^T(\mathbf{S}_3 + \mathbf{Q}_3^T\mathbf{S}_3\mathbf{Q}_3 - \mathbf{Q}_3^T\mathbf{S}_3 - \mathbf{S}_3\mathbf{Q}_3)\mathbf{x} = \sum_{i=N^2-N+1}^{N^2} \delta_i x_i^2 + \sum_{i=1}^{N^2-N} \delta_{i+N}(x_i - x_{i+N)^2} \tag{7.34}$$

where the elements of $\mathbf{S_2}$, $\mathbf{S_3}$, and $\mathbf{S_4}$ have been renamed such that

$$\mathbf{S}_2 = \text{diagonal}(\beta_1, \beta_2, \ldots, \beta_{N^2}) \tag{7.35}$$

$$\mathbf{S}_3 = \text{diagonal}(\gamma_1, \gamma_2, \ldots, \gamma_{N^2}) \tag{7.36}$$

$$\mathbf{S}_4 = \text{diagonal}(\delta_1, \delta_2, \ldots, \delta_{N^2}) \tag{7.37}$$

Since

$$\mathbf{Q} = \sum_{k=1}^{4} [\mathbf{S}_k - \mathbf{S}_k\mathbf{Q}_k - \mathbf{Q}_k^T\mathbf{S}_k + \mathbf{Q}_k^T\mathbf{S}_k\mathbf{Q}_k] \tag{7.38}$$

then $\mathbf{x}^T\mathbf{Q}\mathbf{x} > 0$ for any vector $\mathbf{x}$ as long as all the weighting coefficients are positive. Hence $\mathbf{Q}$ is positive definite.

7.3 Statistical Spatial Approach: MAP Estimation

The above technique tends to smear edges. Statistical techniques however have been successfully used in image processing for edge reconstruction [7, 73, 74, 75, 76]. The original image is modeled as a Markov random field (MRF) [6, 5, 7], and edges are reconstructed by maximum *a posteriori* (MAP) techniques. This is the approach adopted here. Each original frame X and its received version Y are modeled as discrete parameter random fields where each pixel is a continuous random variable. Assuming a prior distribution for $\mathbf{x}$, a maximum *a posteriori* (MAP) estimate is obtained given the received data $\mathbf{y}$. Denoting the estimate of $\mathbf{x}$ by $\hat{\mathbf{x}}$, $\hat{\mathbf{x}} = \arg\max_{\mathbf{x}|\mathbf{y}=\mathbf{D}\mathbf{x}} L(\mathbf{x} \mid \mathbf{y})$, where $L(\mathbf{x} \mid \mathbf{y})$ is the log-likelihood function. In other

words, $L(\mathbf{x} \mid \mathbf{y}) = \ln f(\mathbf{x} \mid \mathbf{y})$, where $f(\mathbf{x} \mid \mathbf{y})$ is the conditional probability density function of $\mathbf{x}$ given $\mathbf{y}$. We shown below that

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x}|\mathbf{y}=\mathbf{D}\mathbf{x}} [-\ln f(\mathbf{x})]. \tag{7.39}$$

Let $\mathcal{A}$ be the event that $\mathbf{y} = \mathbf{D}\mathbf{x}$. In general [77], $f(\mathbf{x}, \mathbf{y})\mathrm{P}(\mathcal{A} \mid \mathbf{x}, \mathbf{y}) = f(\mathbf{x}, \mathbf{y} \mid \mathcal{A})\mathrm{P}(\mathcal{A})$. Using Bayes' rule then,

$$f(\mathbf{x} \mid \mathbf{y})\mathrm{P}(\mathcal{A} \mid \mathbf{x}, \mathbf{y}) = \frac{f(\mathbf{x}, \mathbf{y} \mid \mathcal{A})\mathrm{P}(\mathcal{A})}{f(\mathbf{y})}.$$

But

$$\mathrm{P}(\mathcal{A} \mid \mathbf{x}, \mathbf{y}) = \begin{cases} 0 & \mathbf{y} \neq \mathbf{D}\mathbf{x} \\ 1 & \mathbf{y} = \mathbf{D}\mathbf{x} \end{cases}$$

Thus,

$$f(\mathbf{x} \mid \mathbf{y}) = \frac{f(\mathbf{x}, \mathbf{y} \mid \mathcal{A})\mathrm{P}(\mathcal{A})}{f(\mathbf{y})}.$$

Let $I_{\delta x} = \{\mathbf{z} \mid \|\mathbf{z} - \mathbf{x}\| \leq \delta x\}$, and $I_{\delta y} = \{\mathbf{u} \mid \|\mathbf{u} - \mathbf{y}\| \leq \delta y\}$. Now,

$$f(\mathbf{x}, \mathbf{y} \mid \mathcal{A}) = \lim_{\delta x \to 0} \lim_{\delta y \to 0} \mathrm{P}(\mathbf{x} \epsilon I_{\delta x}, \mathbf{y} \epsilon I_{\delta y} \mid \mathcal{A}).$$

Rewriting $\mathrm{P}(\mathbf{x} \epsilon I_{\delta x}, \mathbf{y} \epsilon I_{\delta y} \mid \mathcal{A})$ as $\mathrm{P}(\mathbf{x} \epsilon I_{\delta x}, \mathbf{y} \epsilon I_{\delta y} \mid \mathcal{A}) = \mathrm{P}(\mathbf{y} \epsilon I_{\delta y} \mid \mathbf{x} \epsilon I_{\delta x}, \mathcal{A})\mathrm{P}(\mathbf{x} \epsilon I_{\delta x} \mid \mathcal{A})$, where

$$\mathrm{P}(\mathbf{y} \epsilon I_{\delta y} \mid \mathbf{x} \epsilon I_{\delta x}, \mathcal{A}) = \begin{cases} 0 & \mathbf{D} I_{\delta x} \cap I_{\delta y} = \phi \\ 1 & \mathbf{D} I_{\delta x} \cap I_{\delta y} \neq \phi \end{cases}$$

and $\mathbf{D} I_{\delta x} = \{\mathbf{D}\mathbf{z} \mid \mathbf{z} \epsilon I_{\delta x}\}$ then, $f(\mathbf{x}, \mathbf{y} \mid \mathcal{A})\mathrm{P}(\mathcal{A}) = f(\mathbf{x} \mid \mathcal{A})\mathrm{P}(\mathcal{A})$. Since

$$f(\mathbf{x} \mid \mathcal{A})\mathrm{P}(\mathcal{A}) = f(\mathbf{x})\mathrm{P}(\mathcal{A} \mid \mathbf{x})$$

then, $L(\mathbf{x} \mid \mathbf{y}) = \ln f(\mathbf{x}) + \ln \mathrm{P}(\mathcal{A} \mid \mathbf{x}) - \ln f(\mathbf{y})$. For a given observed image, the third term in the preceding equation is independent of $\mathbf{x}$, hence the MAP estimate is obtained as

$$\begin{aligned} \hat{\mathbf{x}} &= \arg \max_{\mathbf{x}|\mathbf{y}=\mathbf{D}\mathbf{x}} [\ln f(\mathbf{x}) + \ln \mathrm{P}(\mathcal{A} \mid \mathbf{x})] \\ &= \arg \min_{\mathbf{x}|\mathbf{y}=\mathbf{D}\mathbf{x}} [-\ln f(\mathbf{x}) - \ln \mathrm{P}(\mathcal{A} \mid \mathbf{x})]. \end{aligned}$$

Since,

$$\mathrm{P}(\mathcal{A} \mid \mathbf{x}) = \begin{cases} 0 & \mathbf{y} \neq \mathbf{D}\mathbf{x} \\ 1 & \mathbf{y} = \mathbf{D}\mathbf{x}, \end{cases}$$

then, $\hat{\mathbf{x}} = \arg\min_{\mathbf{x}|\mathbf{y}=\mathbf{D}\mathbf{x}}[-\ln f(\mathbf{x})]$.

Since X is modeled as a Markov Random Field (MRF), the probability density function of $\mathbf{x}$ is given by [5, 7, 78],

$$f(\mathbf{x}) = \frac{1}{Z} \exp\left(-\sum_{c \epsilon C} V_c(\mathbf{x})\right), \tag{7.40}$$

where Z is a normalizing constant known as the *partition function*, $V_c(\cdot)$ a function of a local group of points c known as cliques, and C the set of all cliques [7]. Using Equations (7.39) and (7.40), the MAP estimate of $\mathbf{x}$ is then

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x}|\mathbf{y}=\mathbf{D}\mathbf{x}} \left[\sum_{c \epsilon C} V_c(\mathbf{x})\right].$$

7.3.1 Implementation

The proper choice of V_c is crucial to the reconstruction of the macroblocks. In this case the potential functions are chosen such that

$$\sum_{c \epsilon C} V_c(\mathbf{x}) = \sum_{i=0}^{N_1-1} \sum_{j=0}^{N_2-1} \sum_{m=0}^{3} b_{i,j}^{(m)} \rho(\frac{D_m(X_{i,j})}{\sigma}), \tag{7.41}$$

where $D_0(X_{i,j}) = X_{i,j-1} - X_{i,j}$, $D_1(X_{i,j}) = X_{i-1,j+1} - X_{i,j}$, $D_2(X_{i,j}) = X_{i-1,j} - X_{i,j}$, and $D_3(X_{i,j}) = X_{i-1,j-1} - X_{i,j}$ are used to approximate the first order derivatives at the $i^{th}j^{th}$ pixel. $\rho(\cdot)$ is a cost function, σ a scaling factor, $b_{i,j}^{(m)}$ weighting coefficients, and the set of cliques [7] is $C = \{\ \{(i, j-1), (i, j)\}, \{(i-1, j+1), (i, j)\}, \{(i-1, j), (i, j)\}, \{(i-1, j-1), (i, j)\ \}$.

Several cost functions have been proposed [73, 75]. A convex $\rho(\cdot)$ results in the minimization of a convex functional. The cost function used here is the one introduced by Huber for obtaining robust M-estimates of location [79]. Its advantage is that it is convex, does not heavily penalize edges, and is simpler to implement than most of

the convex cost functions used in the literature [75]. It is defined to be

$$\rho_\gamma(x) = \begin{cases} x^2 & |x| \leq \gamma \\ \gamma^2 + 2\gamma(|x| - \gamma) & |x| > \gamma. \end{cases}$$

Hence, $\sum_{c \epsilon C} V_c(\mathbf{x}) = \sum_{i=0}^{N_1-1} \sum_{j=0}^{N_2-1} \sum_{m=0}^{3} b_{i,j}^m \rho_\gamma(\frac{D_m(X_{i,j})}{\sigma})$. Letting

$$h_\gamma(\mathbf{x}) = \sum_{i=0}^{N_1-1} \sum_{j=0}^{N_2-1} \sum_{m=0}^{3} b_{i,j}^m \rho_\gamma(\frac{D_m(X_{i,j})}{\sigma}),$$

we have

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x}|\mathbf{y}=\mathbf{D}\mathbf{x}} h_\gamma(\mathbf{x}). \tag{7.42}$$

The solution to the equation above can be obtained by means of the iterative conditional modes (ICM) algorithm [80]. In particular if the i^{th} element of $\mathbf{x}$ corresponds to a lost pixel value and $\mathbf{x}_{\partial i}$ denotes the neighborhood of x_i then

$$\hat{x}_i = \arg \max_{x_i} f(x_i \mid \mathbf{x}_{\partial i}). \tag{7.43}$$

Using Equations (7.41) and (7.43), the MAP estimate of pixel (i, j), given its neighbors, is

$$\hat{X}_{i,j} = \arg \min_{X_{i,j}} \sum_{l=i}^{i+1} \sum_{k=j}^{j+1} \sum_{m=0}^{3} b_{l,k}^m \rho_\gamma(\frac{D_m(X_{l,k})}{\sigma}). \tag{7.44}$$

Let $\mathbf{X}_{\partial i}$ denote the neighborhood of the i^{th} macroblock $\mathbf{x_i}$. The MAP estimate of $\mathbf{x_i}$ satisfies $\hat{\mathbf{x}}_\mathbf{i} = \arg \max_\mathbf{x} f(\mathbf{x_i} \mid \mathbf{X}_{\partial i})$. If we let $\mathbf{J_i}$ denote the set of indices of the pixels belonging to $\mathbf{x_i}$, then it can be similarly shown that

$$\hat{\mathbf{x}}_\mathbf{i} = \arg \min_{\mathbf{x_i}} \sum_{(i,j) \epsilon \mathbf{J_i}} \sum_{l=i}^{i+1} \sum_{k=j}^{j+1} \sum_{m=0}^{3} b_{l,k}^m \rho_\gamma(\frac{D_m(X_{l,k})}{\sigma}). \tag{7.45}$$

The solution to Equation (7.45) can be obtained iteratively. This however is computationally intensive. In the subsequent section we describe how we speed up the process of finding a solution by using median filtering techniques to obtain a suboptimal MAP estimate. Note that optimality here does not indicate that the estimate obtained is optimal within the MAP framework and is not a globally optimal estimate.

7.3.2 Median Filtering: A Suboptimal Approach

The choice of γ and σ is crucial to the reconstruction of edges. The smaller the product $\gamma\sigma$, the less the edges are penalized. Since $h_\gamma(\mathbf{x})$ is continuous, convex, and has continuous first partial derivatives, then by successively iterating with respect to each pixel, a global minimum is attained. However, there is more than one global minimum. Using Equation (7.44) we obtain,

$$\begin{aligned}\frac{\partial}{\partial X_{i,j}}h_\gamma(\mathbf{x}) &= \frac{b^0_{i,j+1}}{\sigma}\rho'_\gamma(\frac{D_0(X_{i,j+1})}{\sigma}) + \frac{b^1_{i+1,j-1}}{\sigma}\rho'_\gamma(\frac{D_1(X_{i+1,j-1})}{\sigma}) \\ &+\frac{b^2_{i+1,j}}{\sigma}\rho'_\gamma(\frac{D_2(X_{i+1,j})}{\sigma}) + \frac{b^3_{i+1,j+1}}{\sigma}\rho'_\gamma(\frac{D_3(X_{i+1,j+1})}{\sigma}) \\ &-\sum_{m=0}^{3}\frac{b^m_{i,j}}{\sigma}\rho'_\gamma(\frac{D_m(X_{i,j})}{\sigma}),\end{aligned}$$

where

$$\rho'_\gamma(x) = \begin{cases} 2x & |x| \le \gamma \\ 2\gamma & x > \gamma \\ -2\gamma & x < -\gamma. \end{cases} \tag{7.46}$$

Each pixel in the interior, has 8 neighbors. Let $z_1, z_2, z_3,\ z_4, z_5, z_6, z_7, z_8$ be the 8 neighbors arranged in ascending order, and rename the associated weights $\{b^m_{l,k} \mid l = i, i+1,\ l = j, j+1,\ m = 0 \cdots 3\}$ as $b_1, b_2, b_3,\ b_4, b_5, b_6, b_7, b_8$. Defining

$$\begin{aligned}U &= \{(k,l) \mid k = i-1 \text{ and } l = j-1, j, j+1 \text{ or } k = i \text{ and } l = j-1\}, \\ L &= \{(k,l) \mid k = i+1 \text{ and } l = j-1, j, j+1 \text{ or } k = i \text{ and } l = j+1\},\end{aligned}$$

and

$$\Delta_k(X_{i,j}) = \begin{cases} z_k - X_{i,j} & z_k \epsilon U \\ X_{i,j} - z_k & z_k \epsilon L, \end{cases}$$

then

$$\frac{\partial}{\partial X_{i,j}}h_\gamma(\mathbf{x}) = \frac{1}{\sigma}\sum_{k\epsilon L} b_k \rho'_\gamma(\frac{\Delta_k(X_{i,j})}{\sigma}) - \frac{1}{\sigma}\sum_{k\epsilon U} b_k \rho'_\gamma(\frac{\Delta_k(X_{i,j})}{\sigma}). \tag{7.47}$$

Since, we are iterating for $X_{i,j}$, we need to solve

$$\frac{\partial}{\partial X_{i,j}}h_\gamma(\mathbf{x}) = 0, \tag{7.48}$$

When solving Equation (7.48), three cases need to be considered.

Case1: $|\mathbf{\Delta_k}| \leq \gamma\sigma \ \forall \mathbf{k}$

This occurs when $z_8 - \gamma\sigma \leq z_1 + \gamma\sigma$, and the optimum value of $X_{i,j}$ satisfies the constraint: $|\Delta_k| \leq \gamma\sigma \ \forall k$. Hence, $h_\gamma(\hat{X}_{i,j}) = \sum_{k=1}^{8} b_k(\frac{\Delta_k}{\sigma})^2$. Using Equations (7.48) and (7.46)

$$\hat{X}_{i,j} = \frac{\sum_{k=1}^{8} b_k z_k}{\sum_{k=1}^{8} b_k}. \tag{7.49}$$

Case2: $|\mathbf{\Delta_k}| \leq \gamma\sigma$ for some k

We show here that the optimum estimate $\hat{X}_{i,j}$ satisfies

$$\hat{X}_{i,j} = \frac{\sum_{k=J_1}^{J_2} b_k z_k + \gamma\sigma[\sum_{k=J_2+1}^{8} b_k - \sum_{k=1}^{J_1-1} b_k]}{\sum_{k=J_1}^{J_2} b_k}$$

where J_1 and J_2 satisfy $z_{J_2} - z_{J_1} \leq \gamma\sigma, \ J_2 \geq J_1$.

Assuming positive weights as well as positive values for $z_1 \cdots z_8$ consider the following special case:

Case2A: $|\mathbf{\Delta_k}| > \gamma\sigma \ \forall \mathbf{k}$

In this case the optimal MAP estimate $\hat{X}_{i,j}$ will lie between z_1 and z_8. It will either be within a $\gamma\sigma$ of some z_n or not. If the former is true then, $h_\gamma(\hat{X}_{i,j}) = b_n(\frac{\hat{\Delta}_n}{\sigma})^2 + \sum_{k=1,k\neq n}^{8} b_k[\frac{2\gamma|\hat{\Delta}_k|}{\sigma} - \gamma^2]$ otherwise

$$\begin{aligned} h_\gamma(\hat{X}_{i,j}) &= \sum_{k=1}^{8} b_k[\frac{2\gamma|\hat{\Delta}_k|}{\sigma} - \gamma^2] \\ &= \sum_{k=1}^{8} b_k \frac{2\gamma|\hat{\Delta}_k|}{\sigma} - \gamma^2 \sum_{k=1}^{8} b_k \end{aligned}$$

where

$$\hat{\Delta}_k = \begin{cases} z_k - \hat{X}_{i,j} & z_k \epsilon U \\ \hat{X}_{i,j} - z_k & z_k \epsilon L, \end{cases}$$

Considering the latter, that is $\hat{X}_{i,j} \notin [z_k - \gamma\sigma, z_k + \gamma\sigma] \ \forall k$, let $\tilde{h}_\gamma(X_{i,j}) = \sum_{k=1}^{8} b_k \frac{2\gamma|\Delta_k|}{\sigma}$ For the following intervals and for the variable $X_{i,j}$, $\tilde{h}_\gamma(X_{i,j})$ is a straight line with negative, zero, or positive slope.

1. $X_{i,j} < z_1 - \gamma\sigma$:
   $\tilde{h}_\gamma(X_{i,j}) = \frac{2\gamma}{\sigma}[-(\sum_{k=1}^{8} b_k)X_{i,j} + \sum_{k=1}^{8} b_k z_k]$

2. $z_l - \gamma\sigma < X_{i,j} < z_{l+1} - \gamma\sigma$ for $l = 1 \cdots 7$:
   $\tilde{h}_\gamma(X_{i,j}) = \frac{2\gamma}{\sigma}[(\sum_{k=1}^{l} b_k - \sum_{k=l+1}^{8} b_k)X_{i,j} + \sum_{k=l+1}^{8} b_k z_k - \sum_{k=1}^{l} b_k z_k]$

3. $z_8 - \gamma\sigma < X_{i,j}$:
   $\tilde{h}_\gamma(X_{i,j}) = \frac{2\gamma}{\sigma}[(\sum_{k=1}^{8} b_k)X_{i,j} - \sum_{k=1}^{8} b_k z_k]$.

Since $\tilde{h}_\gamma(X_{i,j})$ is continuous, then there exists an integer J such that $\sum_{k=1}^{J-1} b_k - \sum_{k=J}^{8} b_k < 0$, $\sum_{k=1}^{J} b_k - \sum_{k=J+1}^{8} b_k = 0$, and $\sum_{k=1}^{J+1} b_k - \sum_{k=J+2}^{8} b_k > 0$ that is $\tilde{h}_\gamma(X_{i,j})$ has a flat bottom, or $\sum_{k=1}^{J-1} b_k - \sum_{k=J}^{8} b_k < 0$ and $\sum_{k=1}^{J} b_k - \sum_{k=J+1}^{8} b_k > 0$. If $\tilde{h}_\gamma(X_{i,j})$ does have a flat bottom, then $z_J \le \hat{X}_{i,j} \le z_{J+1}$, otherwise $\hat{X}_{i,j} = z_J$.

Case2B: $|\mathbf{\Delta_k}| \le \gamma\sigma$ for some $\mathbf{k}$

Suppose now that there exist J_1 and J_2 such that $J_1 < J_2$ and $z_{J_2} - z_{J_1} \le \gamma\sigma$, then for $z_{J_1} \le X_{i,j} \le z_{J_2}$,

$$h_\gamma(X_{i,j}) = \sum_{k=1}^{J_1-1} b_k[\frac{2\gamma}{\sigma}|X_{i,j}-z_k|-\gamma^2] + \sum_{k=J_2+1}^{8} b_k[\frac{2\gamma}{\sigma}|X_{i,j}-z_k|-\gamma^2] + \sum_{k=J_1}^{J_2} b_k(\frac{X_{i,j}-z_k}{\sigma})^2.$$

In light of **Case 2A** above, if $\sum_{k=1}^{J_1-1} b_k - \sum_{k=J_1}^{8} b_k > 0$, then there exists an integer J_3 such that $J_3 \le J_1$ and $z_{J_3} \le \hat{X}_{i,j} \le z_{J_3+1}$, or $\hat{X}_{i,j} = z_{J_3}$. Similarly, if $\sum_{k=1}^{J_2} b_k - \sum_{k=J_2+1}^{8} b_k < 0$, then there exists an integer J_4 such that $J_4 \ge J_2$ and $z_{J_4} \le \hat{X}_{i,j} \le z_{J_4+1}$ or $\hat{X}_{i,j} = z_{J_4}$. Otherwise, there exists an integer J_5 such that $J_1 \le J_5 \le J_2$ and $z_{J_5} - \gamma\sigma \le \hat{X}_{i,j} \le z_{J_5} + \gamma\sigma$. Since,

$$\begin{aligned} h_\gamma(X_{i,j}) &= \sum_{k=1}^{J_1-1} b_k[\frac{2\gamma}{\sigma}|X_{i,j}-z_k|-\gamma^2] + \sum_{k=J_2+1}^{8} b_k[\frac{2\gamma}{\sigma}|X_{i,j}-z_k|-\gamma^2] + \sum_{k=J_1}^{J_2} b_k(\frac{X_{i,j}-z_k}{\sigma})^2 \\ &\sum_{k=1}^{J_1-1} b_k[\frac{2\gamma}{\sigma}[X_{i,j}-z_k]-\gamma^2] + \sum_{k=J_2+1}^{8} b_k[\frac{2\gamma}{\sigma}[z_k-X_{i,j}]-\gamma^2] + \sum_{k=J_1}^{J_2} b_k(\frac{X_{i,j}-z_k}{\sigma})^2 \end{aligned}$$

$$\frac{\partial}{\partial X_{i,j}} h_\gamma(\mathbf{x}) = 0, \Rightarrow$$

$$\hat{X}_{i,j} = \frac{\sum_{k=J_1}^{J_2} b_k z_k + \gamma\sigma[\sum_{k=J_2+1}^{8} b_k - \sum_{k=1}^{J_1-1} b_k]}{\sum_{k=J_1}^{J_2} b_k}$$

where J_1 and J_2 satisfy $z_{J_2} - z_{J_1} \le \gamma\sigma$, $J_2 \ge J_1$. As $\gamma\sigma \to \infty$, **Case 2A** above is satisfied, that is, there exits an integer J such that $z_J \le \hat{X}_{i,j} \le z_{J+1}$ when $\sum_{k=1}^{J} b_k = \sum_{k=J+1}^{8} b_k$, or $\hat{X}_{i,j} = z_J$ when $\sum_{k=1}^{J} b_k \ge \sum_{k=J+1}^{8} b_k$. If $J = 3$, then a possible value

for $\hat{X}_{i,j}$, would be the median of $z_1 \cdots z_8$. This is in fact the case when all the weights $b_1 \cdots b_8$ are equal.

Fixing σ and choose γ to be arbitrarily small and positive, then according to **Case 2A**, if equal weights $\{b_{l,k}^m \mid m = 0 \cdots 3,\ l = i, i+1,\ k = j, j+1\}$ are used, one possible choice for the optimum value $\hat{X}_{i,j}$ will be the median of its neighbors, unless there are at least two of the neighboring pixels that are equal in value. Under such conditions, the common pixel value is used. Although this is a suboptimal strategy, the resulting reconstruction technique is faster than searching for the optimum value using line search techniques [81]. The estimate of each missing pixel value is now obtained by finding the median of 8 values instead of performing line search techniques. The advantage of this approach is that a MAP estimate is obtained much faster rendering it attractive for real-time implementation on settop decoder boxes.

7.3.3 Estimation of Boundary Pixels

The above mentioned techniques coupled with macroblock or Slice [24] interleaving are particularly useful in the restoration of intracoded macroblocks that belong to a frame that serves as the anchor frame for a new scene within the same sequence.

An alternative approach that is useful for reconstructing intra coded macroblocks when the current damaged frame and the previous reference frame belong to the same scene is next described. The underlying idea is to try to find a macroblock sized region in the previous frame, X^{-1}, that will maximize the MAP estimate of the boundary pixels of the missing macroblock given its neighbors.

Formally, suppose again that the i^{th} macroblock $\mathbf{x}_i$ is missing. Let $\mathbf{X}_{\partial i}$ denote its neighboring macroblocks and let (m, n) denote the coordinates of the upper left corner of $\mathbf{x}_i$. Establish a search range $\mathcal{S}$ of $(2S+1)\times(2S+1)$ pixels in the previous frame X^{-1} centered at (m, n), that is $\mathcal{S} = \{X_{k,l}^{-1} |\ k\epsilon[m-S, m+S],\ l\epsilon[n-S, n+S],\ k, l \text{ integers}\}$. Let $\mathbf{u}$ denote a macroblock sized region in $\mathcal{S}$, that is $\mathbf{u} \subset \mathcal{S}$, and let $\mathbf{u}_B$ denote the boundary pixels of $\mathbf{u}$, then

$$\hat{\mathbf{x}}_\mathbf{i} = \arg\max_{\mathbf{u}\subset\mathcal{S}} f(\mathbf{u}_B \mid \mathbf{X}_{\partial i}).$$

Using the potential functions described above this can be written as

$$\hat{\mathbf{x}}_{\mathbf{i}} = \arg\min_{\mathbf{u}\subset\mathcal{S}} \sum_{(r,s)|X_{(r,s)}\epsilon\mathbf{u}_B} \sum_{l=r}^{r+1}\sum_{k=s}^{s+1}\sum_{m=0}^{3} b_{l,k}^{m}\rho_{\gamma}(\frac{D_m(X_{l,k})}{\sigma}). \tag{7.50}$$

This technique is particularly useful for restoring I frames that have been heavily damaged due to packet loss.

In the following, we describe how we achieve motion compensated restoration by first estimating the missing motion vectors and then utilizing the estimates of the missing motion vectors to reconstruct the missing macroblocks.

7.4 Temporal Restoration: Motion Vector Estimation

Most frames in an MPEG sequence are predicted frames that have motion vectors associated with their macroblocks by which they are reconstructed at the decoder. A more expedient way of reconstructing a lost macroblock would be to estimate its associated missing motion vector.

Let $\mathbf{v}_i$ be the motion vector associated with the i^{th} macroblock in the current frame. In lossless transmission, $\mathbf{x_i}^{(0)}$, the i^{th} macroblock in the current frame, is reconstructed by the decoder as $\mathbf{x_i}^{(0)} = \mathbf{x}_{\mathbf{i}-\mathbf{v}_i}^{(-1)} + \mathbf{n}_i$. Here $\mathbf{x}_{\mathbf{i}-\mathbf{v}_i}^{(-1)}$ is the macroblock in the reference frame that closely matches $\mathbf{x_i}^{(0)}$, $\mathbf{n}_i$ is the error arising from having replaced $\mathbf{x_i}^{(0)}$ by $\mathbf{x}_{\mathbf{i}-\mathbf{v}_i}^{(-1)}$, and $\mathbf{i}$ indicates the spatial coordinates of the i^{th} macroblock. In lossy transmission it is not possible to recover $\mathbf{n}_i$. The goal is to obtain an estimate for $\mathbf{v}_i$ that will point to $\mathbf{x}_{\mathbf{i}-\mathbf{v}_i}^{(-1)}$.

7.4.1 Deterministic

One approach is to average the motion vectors of surrounding macroblocks [8, 39] and use the average vector to retrieve a version of the lost macroblock. This retrieved macroblock is then averaged with another version obtained via spatial interpolation.

In the following section we present an alternative means of estimating the missing motion vector based on the use of Markov Random Field (MRF) models.

7.4.2 MAP Estimation of Motion Vectors

Utilizing the same MRF model described above to model each component of the motion field we can obtain a MAP estimate of the missing motion vector given its neighboring motion vectors. In addition, using the results of Section 7.3.2 we immediately infer that the median of the motion vectors of the surrounding macroblocks yields a suboptimal estimate of the missing motion vector. These estimates are then utilized to perform motion compensated restoration of the missing macroblock.

It is also evident from Equation (07.49), that when $\sigma = 1$, $\gamma \to \infty$ (larger than the image dimensions suffices), $\mathbf{b}_k = 1$ for $k = 1, \cdots, 8$, and z_ks are motion vectors, the MAP estimate of the motion vector is the average of all surrounding motion vectors as proposed in [8].

7.4.3 Temporal-Spatial Approach

An alternative approach based on using a ternary tree to classify the motion vectors neighboring the missing motion vector and the MRF model of the image and motion fields is described next.

Our approach is based on classifying each neighboring motion vector according to whether each of its components are positive, negative, or zero. This is shown in Figure 7.3 where $\mathbf{v}_x$ denotes the horizontal component of a motion vector and $\mathbf{v}_y$ its vertical component. The idea is to implicitly model the discontinuity in the motion field and is similar to the approaches considered in [82, 83], wherein the discontinuities of motion fields were modeled by means of binary MRFs.

After all the neighboring motion vectors have been classified we then determine the class to which the missing motion vector belongs. This is done by assigning a cost to each class and then choosing the class with the lowest cost. After the class with the lowest cost has been chosen, the motion vectors belonging to that class are modeled via the MRF and a MAP estimate of the missing motion vector obtained. If the MAP estimate is not unique then the motion vector that provides the macroblock with "best" matching boundaries given the neighboring macroblocks is chosen.

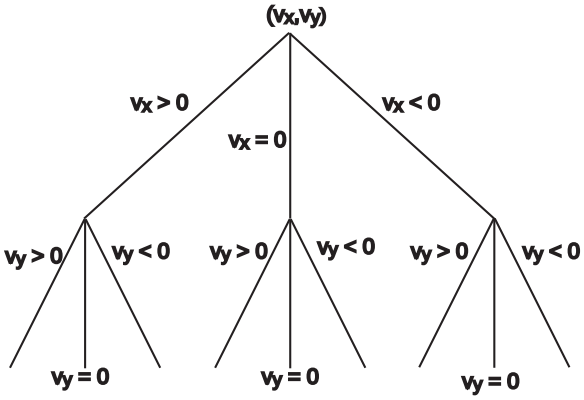


Fig. 7.3. Tree classification of motion vectors.

Formally, let $\mathcal{K} = \{K_i,\ i = 1 \cdots 9\}$ denote the set of 9 classes. Also let $\mathbf{v_t}$, $\mathbf{v_b}$, $\mathbf{v_l}$, and $\mathbf{v_r}$, be the top, bottom, left, and right neighboring motion vectors of the missing motion vector $\mathbf{v}_i$, respectively. Then, the cost C_i incurred by assuming that $\mathbf{v}_i \epsilon K_i$ is given by

$$C_i = \alpha[g_i(\mathbf{v}_i, \mathbf{v_t}) + g_i(\mathbf{v}_i, \mathbf{v_b}) + g_i(\mathbf{v}_i, \mathbf{v_l}) + g_i(\mathbf{v}_i, \mathbf{v_r})]$$

where α is some constant and

$$g_i(\mathbf{v}, \mathbf{u}) = \begin{cases} 0 & \mathbf{v}\epsilon K_i,\ \mathbf{u}\epsilon K_i \\ 1 & \text{otherwise.} \end{cases}$$

That is, $g_i(\cdot, \cdot)$ is 0 when both arguments belong to the same class of motion vectors.

Let $\mathcal{K}^*$ denote the set of classes that have the same minimum cost. It is conceivable that two or more classes will have the same cost. In this case we need to use spatial information to decide between which classes of vectors to choose. However, prior to that we need to obtain a MAP estimate of $\mathbf{v}_i$ given that it belongs to the class $K\epsilon\mathcal{K}^*$. This is done by solving the following for every $K\epsilon\mathcal{K}^*$

$$\{\hat{\mathbf{v}}_i\}_K = \arg\max_{\mathbf{v}\epsilon K} f(\mathbf{v} \mid K)$$

where $\{\hat{\mathbf{v}}_i\}_K$ is the class of vectors that maximize the above equation.

Choosing the motion vector with the "best" matching boundaries is performed as follows. Let $\mathcal{V}$ denote the set of MAP motion vectors, that is $\mathcal{V} = \bigcup_{K\epsilon\mathcal{K}^*}\{\hat{\mathbf{v}}_i\}_K$, then

$\hat{\mathbf{v}}_i$, the estimate to $\mathbf{v}_i$ is given by

$$\hat{\mathbf{v}}_i = \arg\max_{\mathbf{v}\epsilon\mathcal{V}} f(\mathbf{B}_{\mathbf{i-v}}^{(-1)} \mid \mathbf{X}_{\partial i})$$

where $\mathbf{B}_{\mathbf{i-v}}^{(-1)}$ consists of the pixels lying on the boundary of the macroblock $\mathbf{x}_{\mathbf{i-v}}^{(-1)}$, as illustrated in Figure 7.4. Using the potential functions described above this can be rewritten as

$$\hat{\mathbf{v}}_i = \arg\min_{\mathbf{v}\epsilon\mathcal{V}} \sum_{(r,s)|X_{(r,s)}\epsilon\mathbf{B}_{\mathbf{i-v}}^{(-1)}} \sum_{l=r}^{r+1} \sum_{k=s}^{s+1} \sum_{m=0}^{3} b_{l,k}^{m} \rho_\gamma(\frac{D_m(X_{l,k})}{\sigma}). \tag{7.51}$$

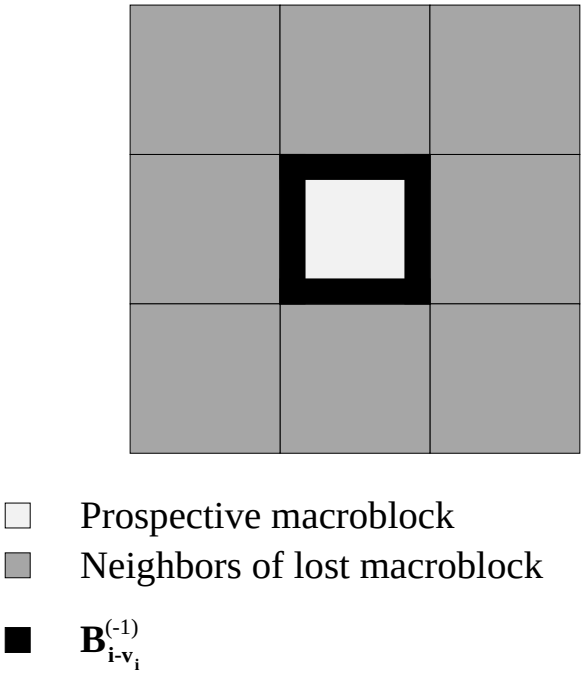


Fig. 7.4. Boundary pixels of prospective macroblock

.

7.4.4 Motion Field Estimation using Gaussian Mixture Models

An alternative to using a ternary tree for classifying the neighboring motion vectors, is to model the motion field via a mixture of 2-D Gaussian probability density functions. The candidates for estimating the missing motion vector are then the means of each probability density function. The motion vector that "best" matches the boundary of the reconstructed macroblock with its neighbors is then chosen.

The number K of the different probability density functions, the means $\{\mu_k,\ k = 1 \cdots K\}$, and the covariance matrices $\{\mathbf{R}_k,\ k = 1 \cdots K\}$ of the probability density functions, are determined via an iterative algorithm, *cluster*[1] [84].

Let π_k denote the probability that a motion vector belongs to the k^{th} mixture or class, and let $\pi = \{\pi_k,\ k = 1 \cdots K\}$, $\mu = \{\mu_k,\ k = 1 \cdots K\}$, $\mathbf{R} = \{\mathbf{R}_k,\ k = 1 \cdots K\}$, and $\theta = (\pi, \mu, \mathbf{R})$. Assuming that there are G available motion vectors, let $V = \{\mathbf{v}_g,\ g = 1 \cdots G\}$.

The following is then performed:

1. K is assigned some value $K^{(0)}$,

2. The parameters $\{\pi_k, \mu_k, \mathbf{R}_k,\ k = 1 \cdots K\}$ are initially assigned the values $\pi_k^{(0)} = \frac{1}{K^{(0)}}$, $\mu_k^{(0)} = y_m$, where $m = \lfloor (k-1)(G-1)/(K^{(0)}-1) \rfloor + 1$, and $R_k^{(0)} = \frac{1}{G}\sum_{g=1}^{G} y_g y_g^T$,

3. At iteration n, the expectation-maximization (EM) algorithm [85] is then used to obtain the parameters $\theta^{(n)}$ that will optimize

$$E\left[\log f(V|K,\theta)|K^{(n-1)}, \theta^{(K^{(n-1)},n-1)}\right] - \frac{1}{2}L\log(2G). \tag{7.52}$$

   Here $f(V|K,\theta)$ is the joint conditional probability density function of the G motion vectors conditioned on that there are K mixtures with parameters specified by θ, and L is the number of continuously valued real numbers required to specify the parameter θ. In this case, $L = K\left(1 + 2 + \frac{(2+1)2}{2}\right) - 1$. Equation(7.52) is known as the minimum description length criterion for estimating the order of a model, and was suggested by Rissanen [86]. The parameters $\theta^{(n)}$ are then stored.

4. If $K^{(n)} > 1$, set $n = n + 1$, $K^{(n+1)} = K^{(n)} - 1$, and goto 3. Otherwise Stop.

Of all the parameters $(K^{(n)}, \theta^{(n)})$, those that minimize

$$\sum_{g=1}^{G} \log\left(\sum_{i=1}^{K} f(\mathbf{v}_g|k,\theta)\pi_k\right) + \frac{1}{2}L\log(2G). \tag{7.53}$$

[1] I wish to express my sincere thanks to Prof. C. E. Bouman for providing the "clustering" software used in this part of the study

are chosen. Here $f(\mathbf{v}_g|k,\theta)$ is the conditional probability density function of the g^{th} motion vector given that it belongs to mixture k. The disadvantage of this technique is that it is computationally intensive but provides a benchmark to which the various other error concealment algorithms are compared.

7.5 Using the Error Concealment Algorithms with the Video Compression Standards

The error concealment algorithms discussed above can be implemented with any of the video compression standards. The only requirement will be that the addresses of the missing macroblocks, as well as the neighboring macroblocks and their motion vectors be provided to the algorithm.

In the case of MPEG-2 sequences, motion compensated error concealment can be achieved by using the temporal-spatial approach or by finding the MAP estimate of the missing motion vector, given its neighbors. Similarly, instead of interpolating the missing low frequency DCT coefficients, spatial reconstruction can be done by using the median filtering approach.

For MPEG-4 sequences, if the decoder cannot use the Motion Boundary Marker (MBM) [67, 68] to properly decode motion vectors, the temporal-spatial approach can be used to estimate them. Similarly, the temporal-spatial approach can be used to perform inter-picture prediction for H.263+ sequences. Again, in all cases the addresses of the missing macroblocks, as well as the neighboring macroblocks and their motion vectors need to be provided

7.6 Results

To test the reconstruction algorithms, the *salesman*, *football*, *flowergarden*, and *hockey* sequences, encoded at data rates of 0.3 Mbits/sec, 1.15 Mbits/sec, 1.5 Mbits/sec, and 1.5 Mbits/sec respectively[2], were multiplexed with MPEG-1 Layer II audio

[2]The GOPs in each sequence were 15 frames, and the frame pattern of each GOP was IBBPBBPBBPBBPBB

streams into MPEG-1 System Layer Streams. For our experiments we assumed that the protocol for transmitting video over networks is the Asynchronous Transfer Mode (ATM) protocol. In ATM, information is transmitted in fixed size packets of data called "cells". The respective System Layer Streams were then packed into ATM cells as described in Chapter 6 and subjected to 2%, 5% and 10% random ATM cell loss.

To determine which macroblocks were lost due to cell loss, the difference between the addresses of the two most recent correctly decoded macroblocks is obtained. This difference between both addresses is taken to be the number of macroblocks (between both macroblocks) that were lost. This can lead to an error in determining which macroblocks in a P or B frame were actually lost or damaged. This is a consequence of the fact that in MPEG video, predicted macroblocks that have zero motion vectors and negligible difference DCT coefficients are not coded but skipped. Under lossless conditions, a macroblock address difference that is greater than 1 is interpreted by the MPEG decoder to mean that the intervening macroblocks are to be duplicated from the reference frame. In particular, those macroblocks in the reference frame that have the same locations as those that are being decoded are used. Thus, our techniques for estimating which macroblocks are missing can misinterpret skipped macroblocks as being lost. This may result in skipped macroblocks being improperly constructed particularly if all the surrounding macroblocks have non-zero motion vectors.

Having determined the missing macroblocks, they are either delineated by assigning them the values (128,0,0) in the YUV space, or they are reconstructed. The delineation serves to indicate which macroblocks have been affected, as well as to show the effect error propagation on the quality of the decoded images. For instance, the damage to an I frame will propagate to the rest of the frames within the GOP to which the I frame belongs.

Having determined the missing macroblocks, two different decoded sequences are generated. In one sequence the pixels belonging to the missing macroblock are assigned by the decoder the values (128,0,0), in the YUV space. This indicates which

macroblocks have been affected by packet loss, and shows the effect of error propagation on the quality of the decoded images. The damage in an I frame, for instance, will propagate to the rest of the frames within that GOP to which the I frame belongs. Subsequent P and B frames will display these values. The second sequence consists of a decoded sequence in which the macroblocks have been restored.

Reconstruction proceeds as follows:

- If the damaged frame is an I frame and all the neighbors of a missing macroblock are available, then reconstruction is performed by means of the spatial reconstruction techniques discussed above.
- If the first I frame in the sequence is damaged, then spatial reconstruction technique based on median filtering is used.
- If the damaged frame is an I frame and some of the neighbors of a missing macroblock are not available, then reconstruction is performed by searching for the macroblock in the most recent I or P frame that optimizes the boundary pixels (Equation (7.50)). The search space used is 21×21 pixels in size.
- If the damaged frame is a P or B frame, reconstruction as follows:
  - If the missing macroblock is surrounded by intracoded macroblocks it is then reconstructed by the same method used for restoring macroblocks missing from an I frame,
  - otherwise the missing macroblock is assumed to be intercoded and it is reconstructed by first estimating its associated motion vector. This is done by a number of ways enumerated below:
    1. Temporal replacement, that is a motion vector with zero components is used.
    2. The average of the surrounding motion vectors is obtained.
    3. The median of the surrounding motion vectors is obtained.

4. The MAP estimate of the missing motion vector given its neighboring motion vectors is obtained.
5. The Temporal-Spatial approach is used to estimate the missing motion vector.
6. The Gaussian Mixture model is used to estimate the missing motion vector.

The estimate of the missing motion vector is then used to provide error concealment. This is achieved by replacing the missing macroblock by the region in the past I or P reference frame to which the estimated motion vector is pointing. For cases 1-5, if the macroblock to be reconstructed has lost all its neighbors, it is replaced by the macroblock in the previous reference frame that has the same location, i.e. temporal replacement is used. A comparison of the performance of the different techniques will be given later.

For all of the motion compensated techniques, the parameters σ, γ, $\{\mathbf{b}_k,\ k = 1, \cdots, 8\}$ were set to unit value. It was observed that changing the values of these parameters did not significantly impact the quality of the reconstruction.

Figure 7.5a is a decoded frame from the *salesman* sequence. Shown in Figure 7.5b is the same frame with some of its macroblocks missing. In Figure 7.5c the deterministic spatial interpolation algorithm (Section 7.2.2) was used to reconstruct the missing data. The reconstruction peak-signal-to-noise ratio (PSNR) was found to be 36.60 dB. The PSNR value was obtained via

$$\text{PSNR} = 10 \log \frac{255^2}{MSE(Y) + MSE(U) + MSE(V)}$$

where $MSE(\cdot)$ denotes the mean square error of the reconstructed color component.

Figure 7.6a is a decoded frame from the same sequence, and Figure 7.6b is the same frame but some of its macroblocks are missing due to cell loss. In Figure 7.6c the deterministic spatial interpolation algorithm of Section 7.2.1 was also applied to

reconstruct the missing data. As evident the reconstruction algorithm performed well except in the case of edges which were smeared. The advantage to such a technique is its simplicity, speed and the fact that it is performed in the spatial domain unlike the method proposed in [48, 49, 46] which reconstructs lost DCT coefficients.

It was observed in general that the both deterministic techniques had comparable performance.

In Figures 7.7a and 7.7b we show the original and damaged frames (due to missing macroblocks), from the *salesman* sequence. Reconstruction was performed spatially in Figure 7.7c via median filtering, in Figure 7.7d by iteratively solving for the MAP estimate of each missing pixel within the damaged macroblock, and in Figure 7.7e by means of the spatial technique of Section 7.2.2. The reconstruction PSNRs were found to be 28.36 dB, 28.14406 dB, and 27.99 dB respectively. From a PSNR point of view the performance of all techniques is comparable. It was observed that the reconstructions due to the first two techniques to have sharper edges. Furthermore, our approach based on median filtering is attractive since it can be implemented in real-time [87], does not require a search for dominant edges, and operates on spatial data rather than DCT coefficients which may not be available.

In our experiments it was observed that using values for σ that are greater than one and $\gamma = 1.0$ provided the best MAP restoration when the weights $\{\mathbf{b}_k\ , k = 1 \cdots 8\}$ were of unit value. This can be seen since for $\gamma = 1.0$, $\sigma > 1$ values will not heavily penalize edges. The image in Figure 7.7d was reconstructed with $\sigma = 100$. It was also observed that using the median values of the border pixels as initial values led to rapid convergence to the optimal MAP estimate.

Figure 7.8a is a decoded frame from the *salesman* sequence. Due to random cell loss major portions the frame were lost as shown in Figure 7.8b. Reconstruction is performed by searching for the motion vector that minimizes Equation (7.50). The search space for the motion vectors did not exceed an area of 7×7 pixels and hence an exhaustive search for the motion vector was implemented. A small search region was used in this case since there is little motion in this sequence. In the case of sequences

where a substantial amount of motion exists an exhaustive search may be too costly and thus other searches such as the logarithmic search may be implemented but at a cost of lower fidelity. As seen the reconstructed version in Figure 7.8c closely matches the original in Figure 7.8a with a reconstruction PSNR value of 35.72 dB.

A hybrid of temporal and spatial interpolation was also performed. Figure 7.9b, is a frame from the the *salesman* sequence that has also been corrupted due to cell loss and Figure 7.9c is the restored version based on minimizing Equation (7.50). In Figure 7.9d the statistical spatial interpolation algorithm was applied to Figure 7.9c for further restoration.

Figure 7.10a is a frame from the *flowergarden* sequence, and Figure 7.10b is the same frame with missing macroblocks due to 5% ATM cell loss. The frame is restored via temporal replacement, obtaining the average of the neighboring motion vectors, finding the median of the neighboring motion vectors, obtaining the MAP estimate of the missing motion vector, via the temporal-spatial approach, and using the Gaussian mixture Model in Figures 7.10c, 7.10d, 7.11a, 7.11b, and 7.11c respectively. The reconstruction PSNR values are 26.76 dB, 27.51 dB, 28.78 dB, 30.19 dB, 30.30 dB, and 28.89 dB respectively.

As is evident, the use of a MRF field that does not penalize the discontinuities in the motion field outperforms using the average or the median of the surrounding motion vectors. It is also observed that using spatial information, as is done in the Temporal-Spatial approach, results in better restoration. This can be seen in Figure 7.11c where it is evident that the damaged portions of the tree trunk are almost perfectly restored. This is attributed to the fact that the Temporal-Spatial approach attempts to preserve the discontinuities in the motion field while matching the macroblock boundaries. In this case, the "best" matching boundaries where those that preserved the discontinuity between the tree trunk and the background. Also shown in Figures 7.10c, 7.10d, 7.11a, 7.11b, and 7.11c are the effect of error propagation due to inaccurate restoration.

For comparison purposes we provide the reconstruction PSNR values for the various error concealment schemes at 2%, 5% and 10% random ATM cell loss, for the *flowergarden*, *football* and *hockey* sequences in Figures 7.12 - 7.29, respectively. We also provide the average PSNR values for the different error concealment strategies at the same ATM cell loss rates for the same three sequences in Tables 7.1, 7.2, and 7.3 respectively.

Table 7.1 Average PSNR values in dB for the different error concealment schemes for the *flowergarden*, *football*, and *hockey* sequences at a 2% ATM cell loss rate.

Error Concealment Technique	*flowergarden*	*football*	*hockey*
Temporal Replacement	32.87	33.98	37.27
Average Motion Vector	35.57	34.54	39.53
Median Motion Vector	35.99	35.09	40.46
MAP estimation of Motion Vector	36.58	35.80	41.21
Temporal-Spatial Approach	36.75	35.80	41.87
Gaussian Mixture	35.13	35.26	41.01

In general, restoration based on finding the MAP estimate of the missing motion vector or using the Temporal-Spatial approach was better than that attained by the other techniques by at least 1 dB. Furthermore, the gap in performance between the Temporal-Spatial approach and the MAP estimation of the missing motion vector widened in the case of both the *football* and *hockey* sequences. This is also seen in Figures 7.20 and 7.26, and can be attributed to the fact that motion field of *flowergarden* is uniform, unlike that of *football* and *hockey*. In the case of the latter, the motion vectors do not point in one general direction. In addition, it is possible for a macroblock to lie on the boundary between two objects moving in opposite directions. In such a case the use of spatial data was needed to determine which object the missing macroblock belonged to. It is also observed that using the median of the neighboring motion vectors performed better than using the average of the neighboring motion vectors.

Table 7.2 Average PSNR values in dB for the different error concealment schemes for the *flowergarden*, *football*, and *hockey* sequences at a 5% ATM cell loss rate.

Error Concealment Technique	*flowergarden*	*football*	*hockey*
Temporal Replacement	30.12	31.51	34.94
Average Motion Vector	32.24	32.41	36.67
Median Motion Vector	32.56	32.83	37.32
MAP estimation of Motion Vector	33.44	33.10	38.10
Temporal-Spatial Approach	33.48	33.60	39.28
Gaussian Mixture	32.28	32.51	38.34

We also simulated ATM cell loss by simulating an ATM switch with a finite buffer and fixed service as shown in Figure 7.30. ATM cells arriving to the switch were served on a first come first serve basis, unless they had to be dropped to make room for incoming higher priority cells when the buffer was full. The input stream consisted of cells carrying data belonging to various MPEG-1 System Layer Streams. The cell arrival rate was governed by the date rate of each System Layer Stream. Each of the em flowergarden, *football*, and *hockey* sequences were multiplexed with 25 other MPEG-1 System Layer Streams streams carrying video data coded at a rate of 1.5 Mbits/sec were used. A buffer size of 5000 cells was used, and the service rate was adjusted such that 0.2%, 0.5%, and 1% of the cells carrying the data belonging to *flowergarden*, *football*, or *hockey* were dropped. These loss rates were chosen since they are more close to the actual loss rates of actual ATM networks. The initial arrival times of all the sequences were randomly chosen.

Figure 7.31a is a frame from the *flowergarden* sequence, and Figure 7.31b is the same frame with missing macroblocks due to 1% ATM cell loss. The frame is restored via temporal replacement, obtaining the average of the neighboring motion vectors, finding the median of the neighboring motion vectors, obtaining the MAP estimate of the missing motion vector, via the temporal-spatial approach, and using the Gaussian mixture Model in Figures 7.31c, 7.31d, 7.32a, 7.32b, and 7.32c respectively. The

Table 7.3 Average PSNR values in dB for the different error concealment schemes for the *flowergarden*, *football*, and *hockey* sequences at a 10% ATM cell loss rate.

Error Concealment Technique	*flowergarden*	*football*	*hockey*
Temporal Replacement	27.69	28.76	31.73
Average Motion Vector	29.88	29.44	33.45
Median Motion Vector	30.19	29.79	33.87
MAP estimation of Motion Vector	30.52	30.29	34.66
Temporal-Spatial Approach	33.06	30.29	35.26
Gaussian Mixture	29.05	29.66	34.97

reconstruction PSNR values are 32.36 dB, 32.36 dB, 32.36 dB, 32.89 dB, 32.09 dB, and 32.36 dB respectively.

For comparison purposes we provide the reconstruction PSNR values for the various error concealment schemes for the same three sequences at cell loss rates of 0.2%, 0.5%, and 1% in Figures 7.33 - 7.50, respectively. We also provide the average PSNR values for the different error concealment strategies at 0.2%, 0.5%, and 1% ATM cell loss rates for the same three sequences in Tables 7.4, 7.5, and 7.6, respectively.

It was observed that in all cases, I frames sustained most of the damage due to ATM cell loss. This is due to the fact that I frames generate more data than P and B frames, and hence cells transporting I frame data have a larger arrival rate than those carrying P and B frame data. It was observed that a lesser number of P frames were damaged in the case of *flowergarden* than in the case of *football* and *hockey*. This is due to the fact that the motion field of *flowergarden* is more uniform than that of *football* and *hockey*. In addition it was also observed that P frames *football* and *hockey* had more intracoded macroblocks than in the case of *flowergarden*. Thus, the intercoded frames belonging to *football* and *hockey* had higher data rates than those of *flowergarden*. This resulted in the simulated ATM switch relieving buffer overflow by dropping ATM cells carrying data belonging to intercoded frames in addition to those carrying data belonging to intracoded frames.

As shown in Figures 7.33 - 7.38, the performance of all techniques is comparable in the case of the *flowergarden* sequence. It is also observed that using temporal replacement, the average of the motion vectors, and the median of the motion vectors resulted in the same reconstruction. This is due to the fact that only I frames that sustained damage. Furthermore, the damaged areas were mostly contiguous regions of macroblocks. Thus, the error concealment techniques had to resort to temporal replacement most of the time. This however, as shown in Figures 7.39 - 7.50 is not the case for the *football* and *hockey* sequences where all frame types were affected by cell loss. In this case, the different algorithms have different performances. It is observed that the Temporal-Spatial approach did result in better reconstruction, especially in the case of the *hockey* sequence, since a greater number of intercoded frames had been damaged. We thus conclude that our Temporal-Spatial approach will provide better error concealment than the other techniques when both intracoded and intercoded frames have been damaged. Furthermore, modeling the motion fields of the various images as mixture of multivariate Gaussian Random Variables offers no significant advantage from a performance as well as computation perspective. It is to be noted that our simulations do not realistically simulate actual networks, since we only considered traffic consisting of only MPEG-1 System Layer sequences.

It is to be noted that all of the above simulations, the corrupted images obtained from the unrestored sequences contain

Table 7.4 Average PSNR values in dB for the different error concealment schemes for the *flowergarden*, *football*, and *hockey* sequences when 0.2% of the ATM cells are dropped due to buffer overflow.

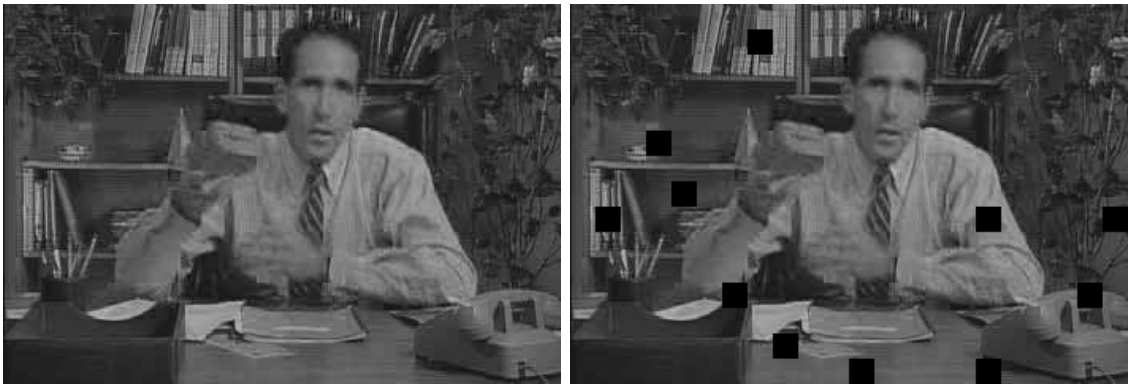
Error Concealment Technique	*flowergarden*	*football*	*hockey*
Temporal Replacement	37.74	40.78	42.24
Average Motion Vector	37.74	40.37	46.52
Median Motion Vector	37.74	40.88	49.91
MAP estimation of Motion Vector	37.21	41.52	49.34
Temporal-Spatial Approach	37.44	41.52	50.92
Gaussian Mixture	37.74	41.39	47.95

Table 7.5 Average PSNR values in dB for the different error concealment schemes for the *flowergarden*, *football*, and *hockey* sequences when 0.5% of the ATM cells are dropped due to buffer overflow.

Error Concealment Technique	*flowergarden*	*football*	*hockey*
Temporal Replacement	34.18	37.56	40.47
Average Motion Vector	34.18	38.91	43.45
Median Motion Vector	34.18	39.36	45.54
MAP estimation of Motion Vector	34.20	39.32	45.42
Temporal-Spatial Approach	34.13	39.54	45.96
Gaussian Mixture	34.15	38.77	44.60

Table 7.6 Average PSNR values in dB for the different error concealment schemes for the *flowergarden*, *football*, and *hockey* sequences when 1% of the ATM cells are dropped due to buffer overflow.

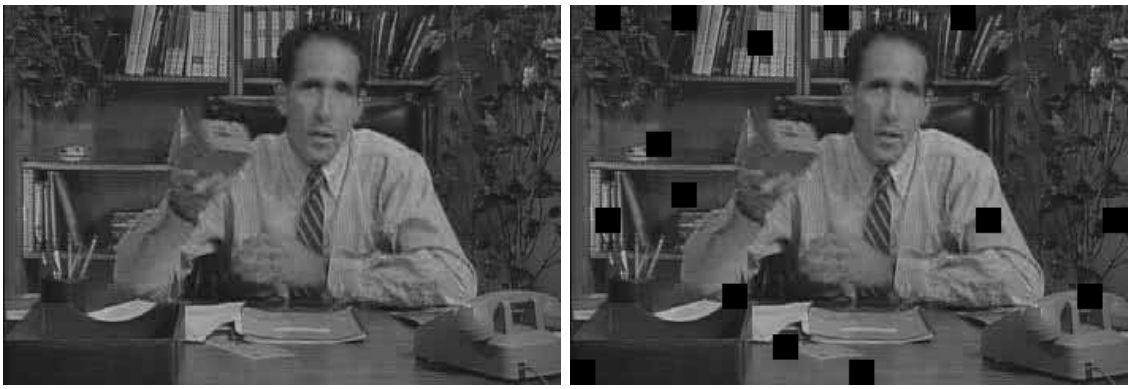
Error Concealment Technique	*flowergarden*	*football*	*hockey*
Temporal Replacement	33.90	35.01	38.90
Average Motion Vector	33.90	35.13	40.83
Median Motion Vector	33.90	35.99	41.86
MAP estimation of Motion Vector	33.60	36.02	42.30
Temporal-Spatial Approach	33.60	36.00	42.80
Gaussian Mixture	33.89	35.76	41.43



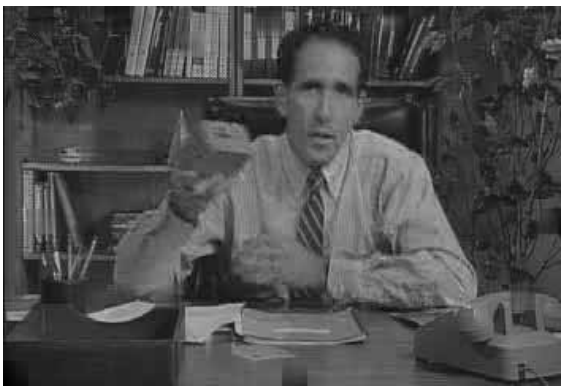
(a) (b)

(c)

Fig. 7.5. Deterministic spatial interpolation of lost data. The figure in (a) is a decoded frame from the *salesman* sequence. Depicted in (b) is a version with randomly missing macroblocks. In (c) the restoration is based on using the deterministic spatial approach Section 7.2.2.

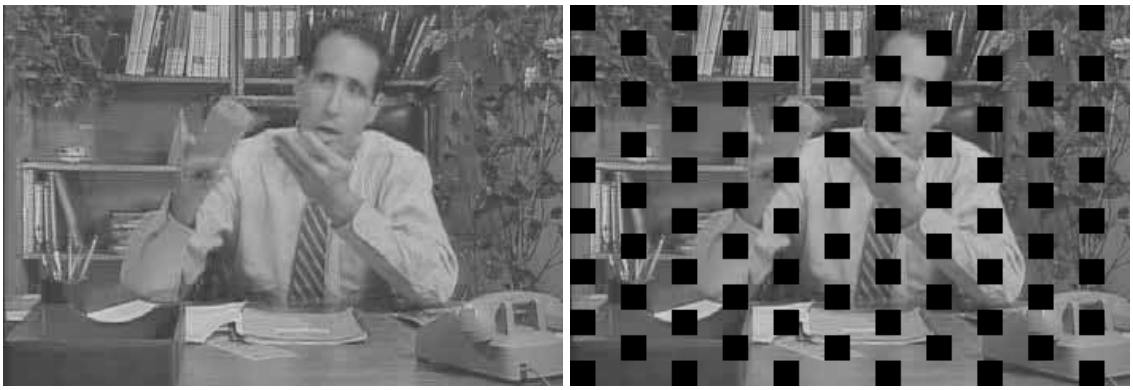


(a) (b)



(c)

Fig. 7.6. Deterministic spatial interpolation of lost data. The figure in (a) is a decoded frame from the *salesman* sequence. Depicted in (b) is a version with randomly missing macroblocks. These were reconstructed spatially by means of the technique described in Section 7.2.1.



(a) (b)

(c) (d)

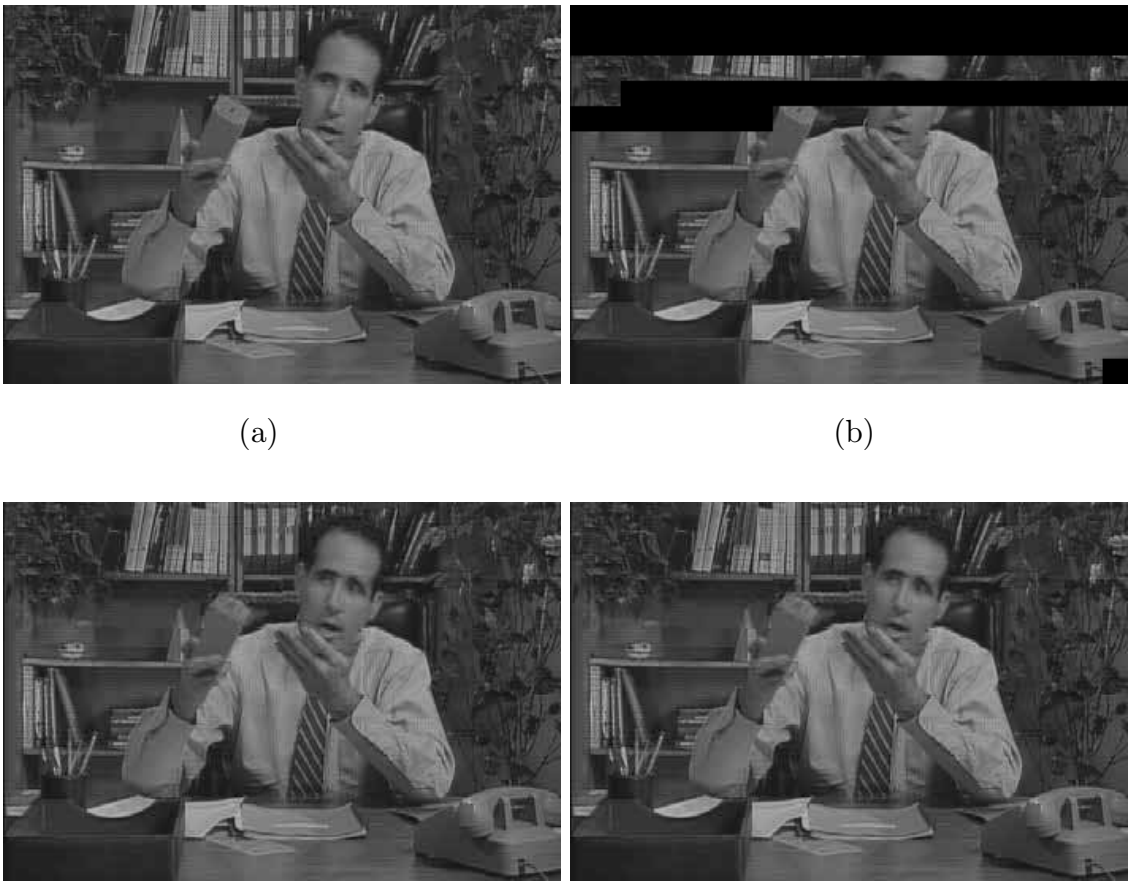
(e)

Fig. 7.7. Spatial reconstruction. (a) is a decoded frame from the *salesman* sequence, in (b) it is missing macroblocks, in (c) it is reconstructed using median filtering, in (d) it is reconstructed by using line search techniques to obtain the MAP estimates with $\sigma = 100.0$, $\gamma = 1.0$, $\mathbf{b}_k = 1.0$ for $k = 1 \cdots 8$, and in (e) it is reconstructed using the technique described in Section 7.2.2.

(a) (b)

(c)

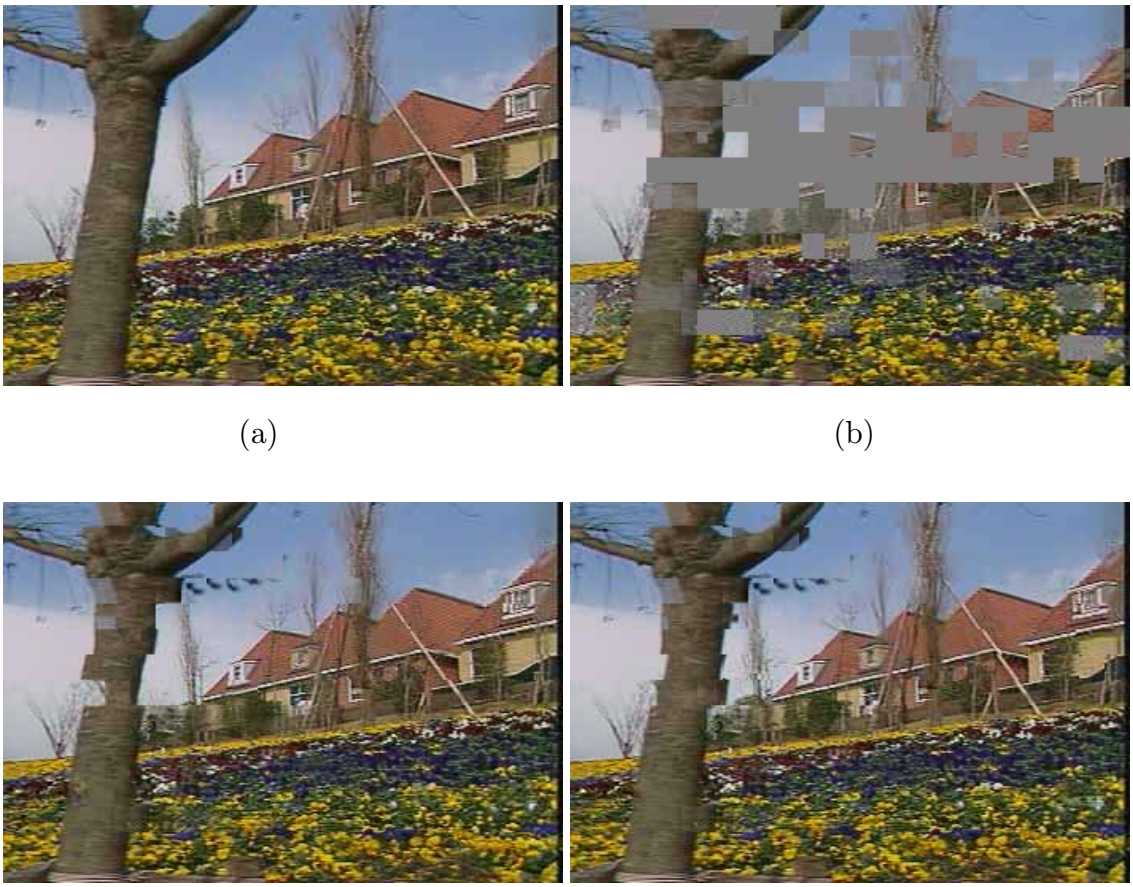
Fig. 7.8. Reconstruction based on the MRF model of the frames. The figure in (a) is a decoded frame from the *salesman* sequence, and that in (b) is a damaged version due to ATM cell loss. The reconstructed version, shown in, (c), was obtained minimizing Equation 7.50.



(a) (b)

(c) (c)

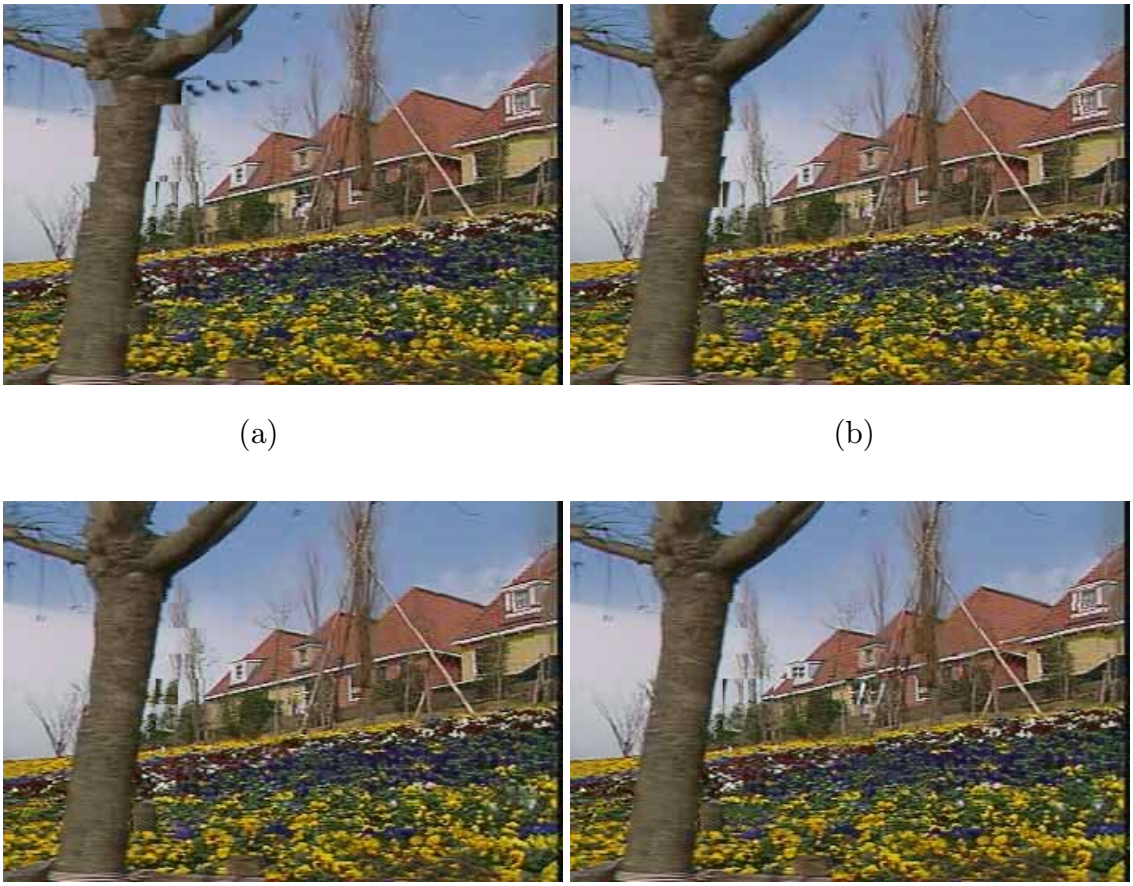
Fig. 7.9. Reconstruction based on the MRF model followed by smoothing of reconstructed data using MAP estimation. Shown in (a) is a decoded frame from the *salesman* sequence. The frame was damaged due to ATM cell loss, shown in (b), and reconstructed in (c) and (d). In (c) the frame was restored by minimizing Equation 7.50. Shown in (d) is the outcome of having applied the statistical spatial technique for reconstruction to the image in (c).



(a) (b)

(c) (d)

Fig. 7.10. (a): decoded frame from the *flowergarden* sequence, (b): frame is damaged due to 5% ATM cell loss, (c): the frame was restored by using temporal replacement, (d): the frame was reconstructed by finding the average of the neighboring motion vectors. The PSNR values are 26.76 dB and 27.51 dB respectively.



(a) (b)

(c) (d)

Fig. 7.11. (continuation of previous figure) (a): the frame was restored by finding the median of the neighboring motion vectors, (b): the frame was reconstructed by finding the MAP estimate of the missing motion vector, (c): the frame was restored by using the temporal-spatial approach, and (d): the frame was reconstructed using the Gaussian mixture model. The PSNR values are 28.78 dB, 30.19 dB, 30.30 dB, and 28.89 dB respectively.

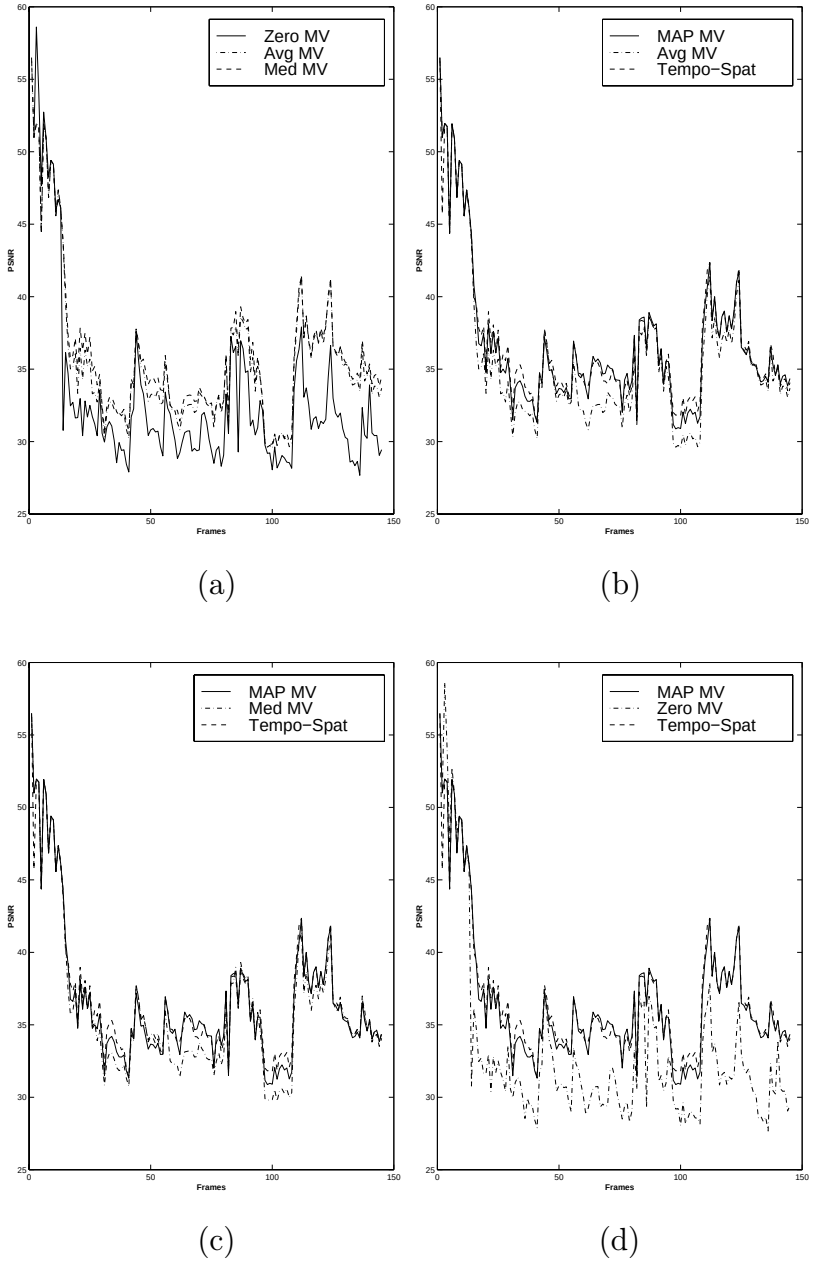


Fig. 7.12. Reconstruction PSNR values for the *flowergarden* sequence when 2% of the cells were dropped. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

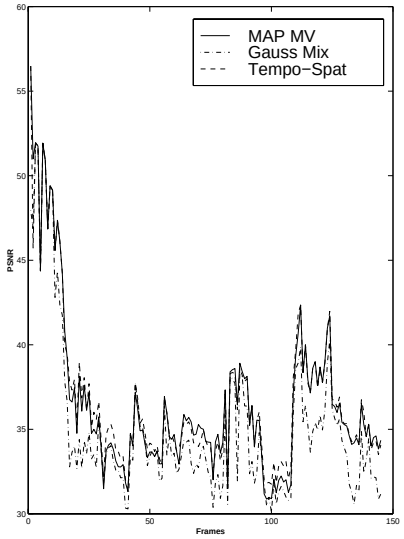


Fig. 7.13. (continuation of the previous figure) Reconstruction PSNR values for the *flowergarden* sequence when 2% of the cells were dropped: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

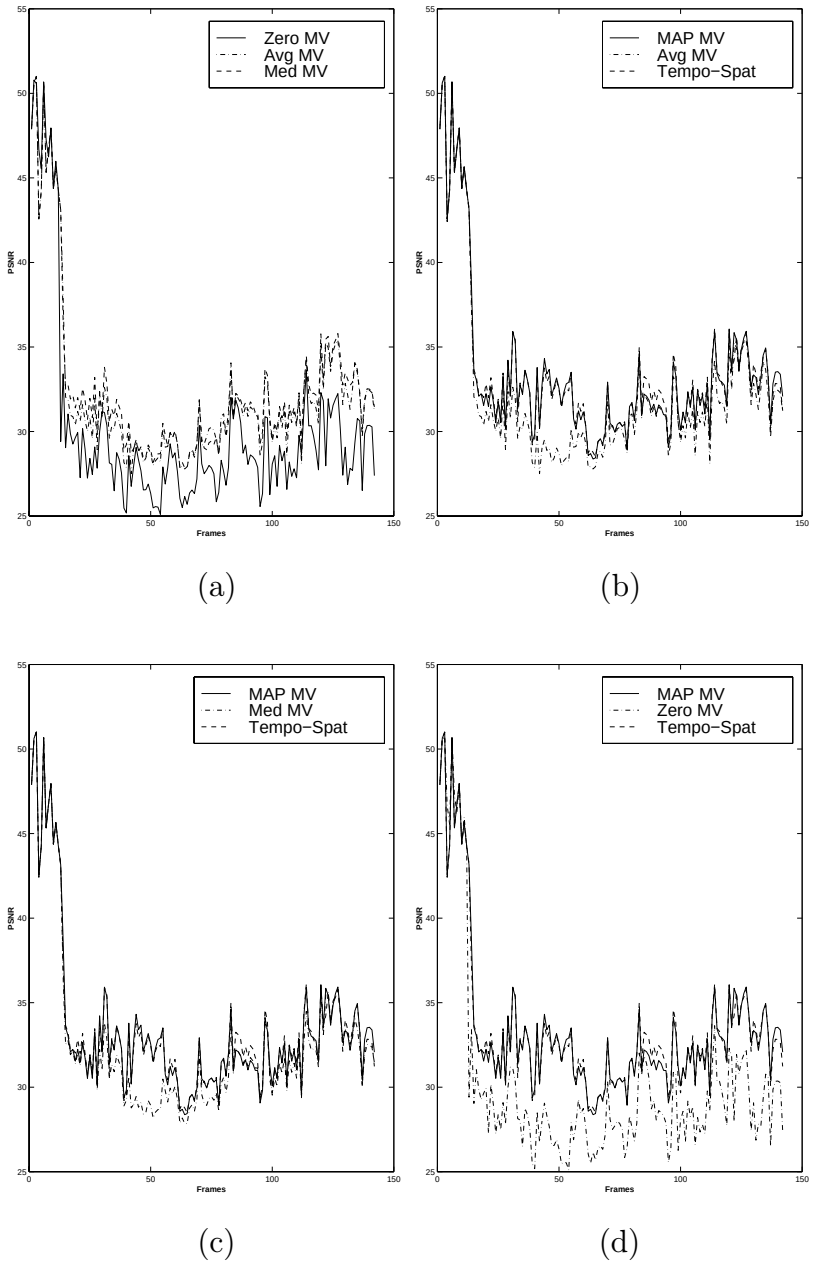


Fig. 7.14. Reconstruction PSNR values for the *flowergarden* sequence when 5% of the cells were dropped. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

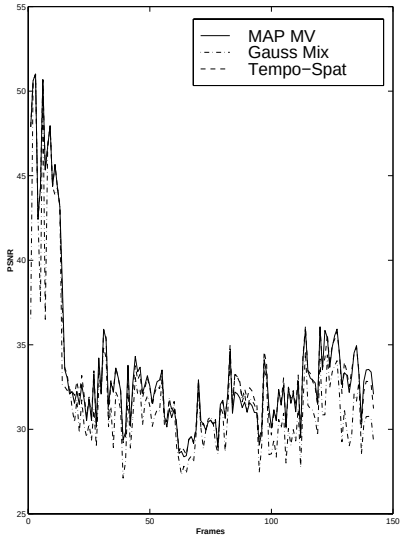


Fig. 7.15. (continuation of the previous figure) Reconstruction PSNR values for the *flowergarden* sequence when 5% of the cells were dropped: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

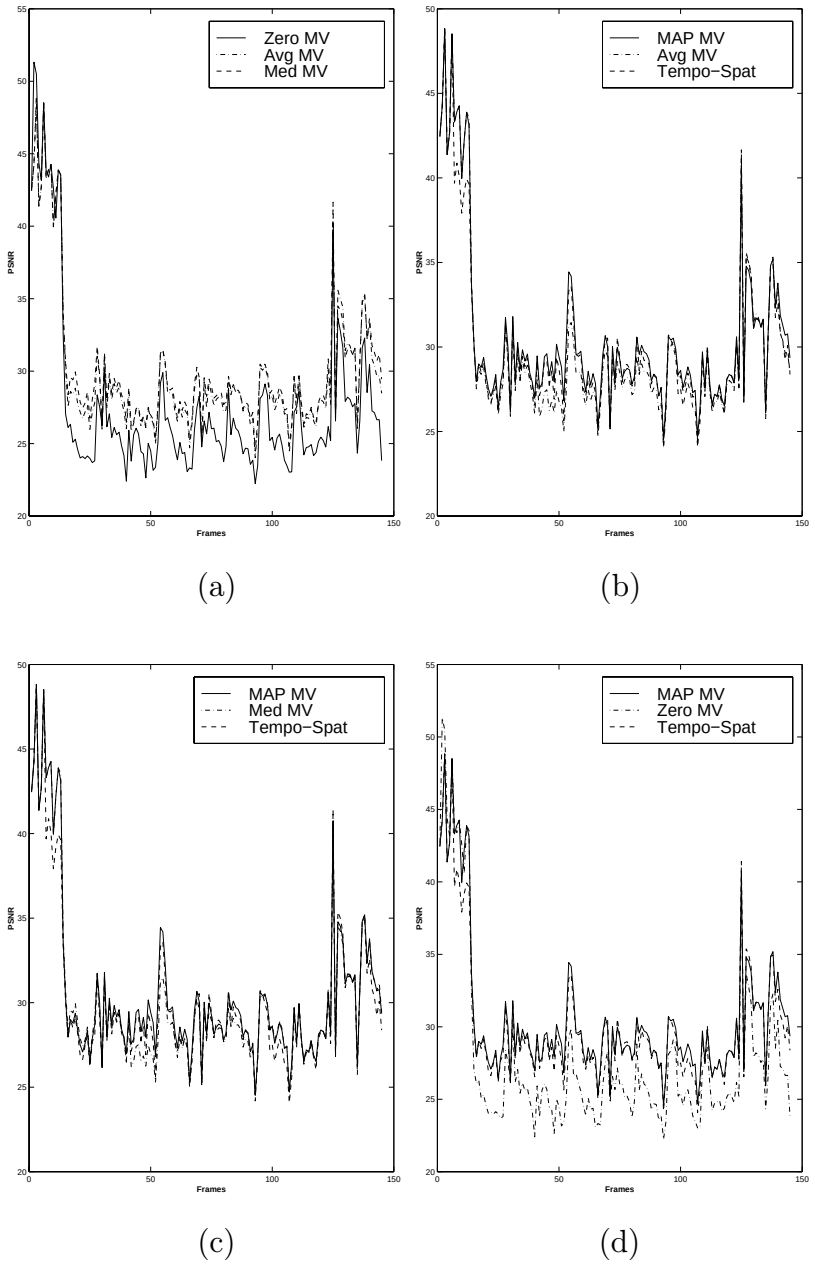


Fig. 7.16. Reconstruction PSNR values for the *flowergarden* sequence when 10% of the cells were dropped. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

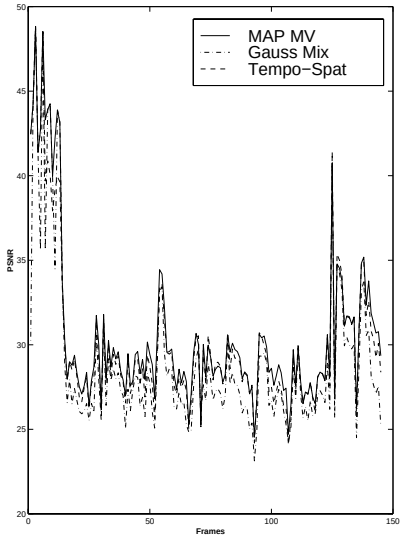


Fig. 7.17. (continuation of the previous figure) Reconstruction PSNR values for the *flowergarden* sequence when 10% of the cells were dropped: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

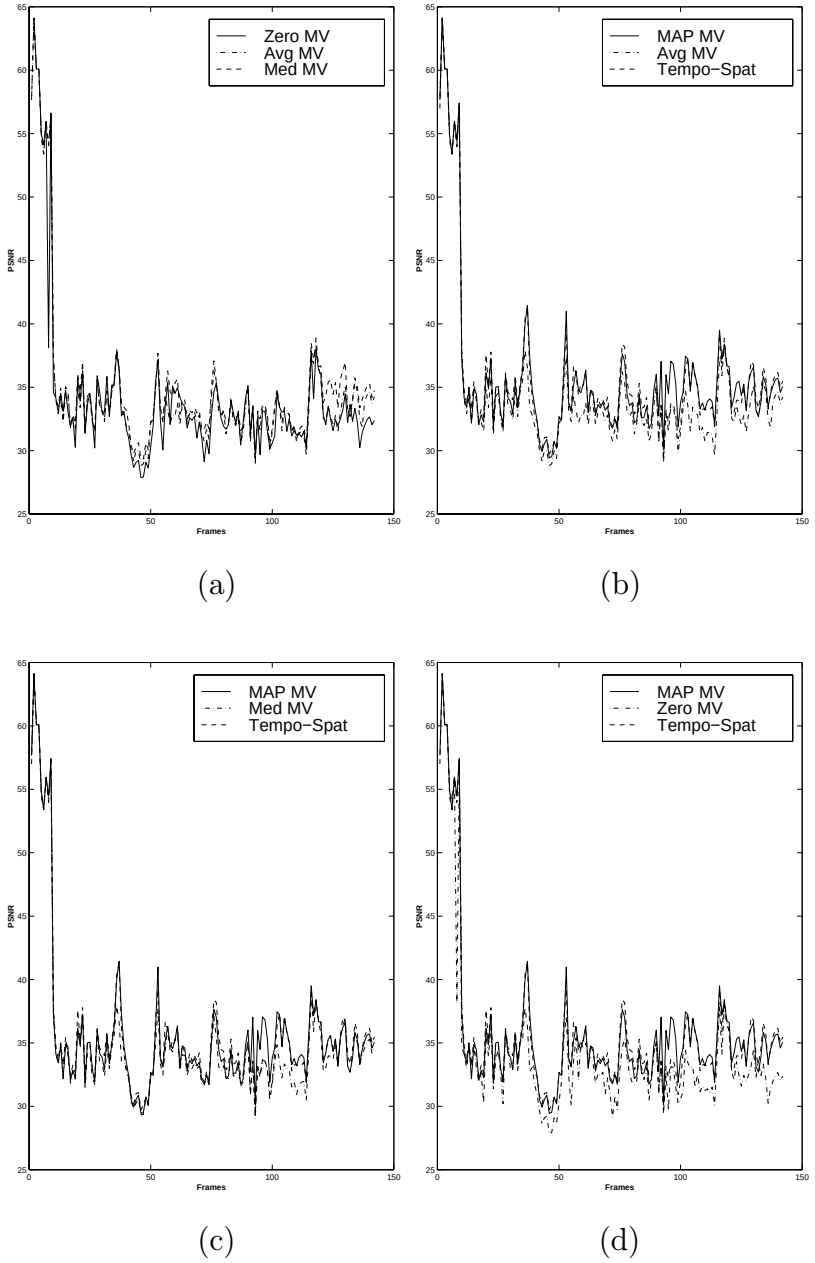


Fig. 7.18. Reconstruction PSNR values for the *football* sequence when 2% of the cells were dropped. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

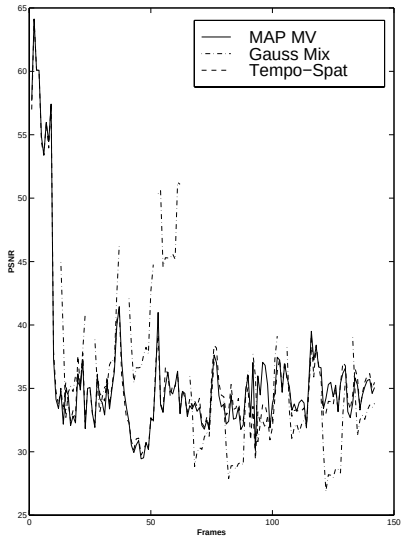


Fig. 7.19. (continuation of the previous figure) Reconstruction PSNR values for the *football* sequence when 2% of the cells were dropped: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

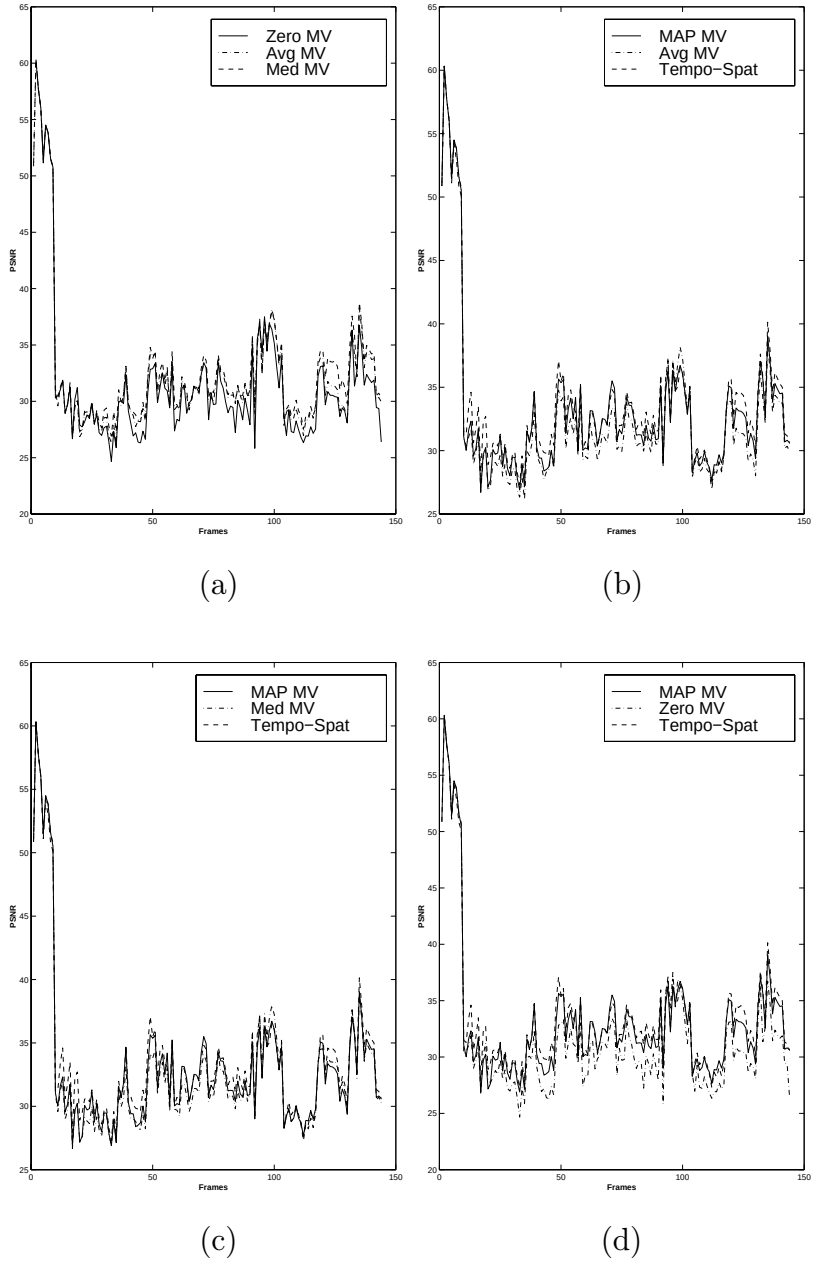


Fig. 7.20. Reconstruction PSNR values for the *football* sequence when 5% of the cells were dropped. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

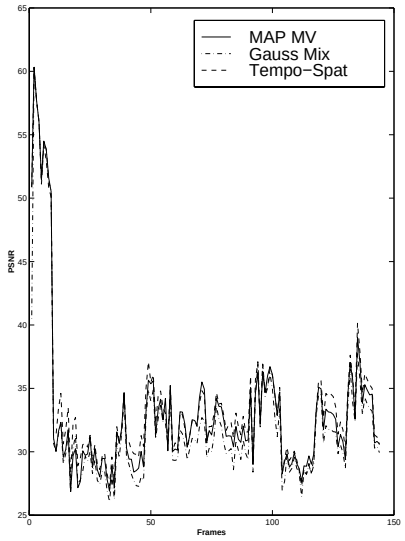


Fig. 7.21. (continuation of the previous figure) Reconstruction PSNR values for the *football* sequence when 5% of the cells were dropped: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

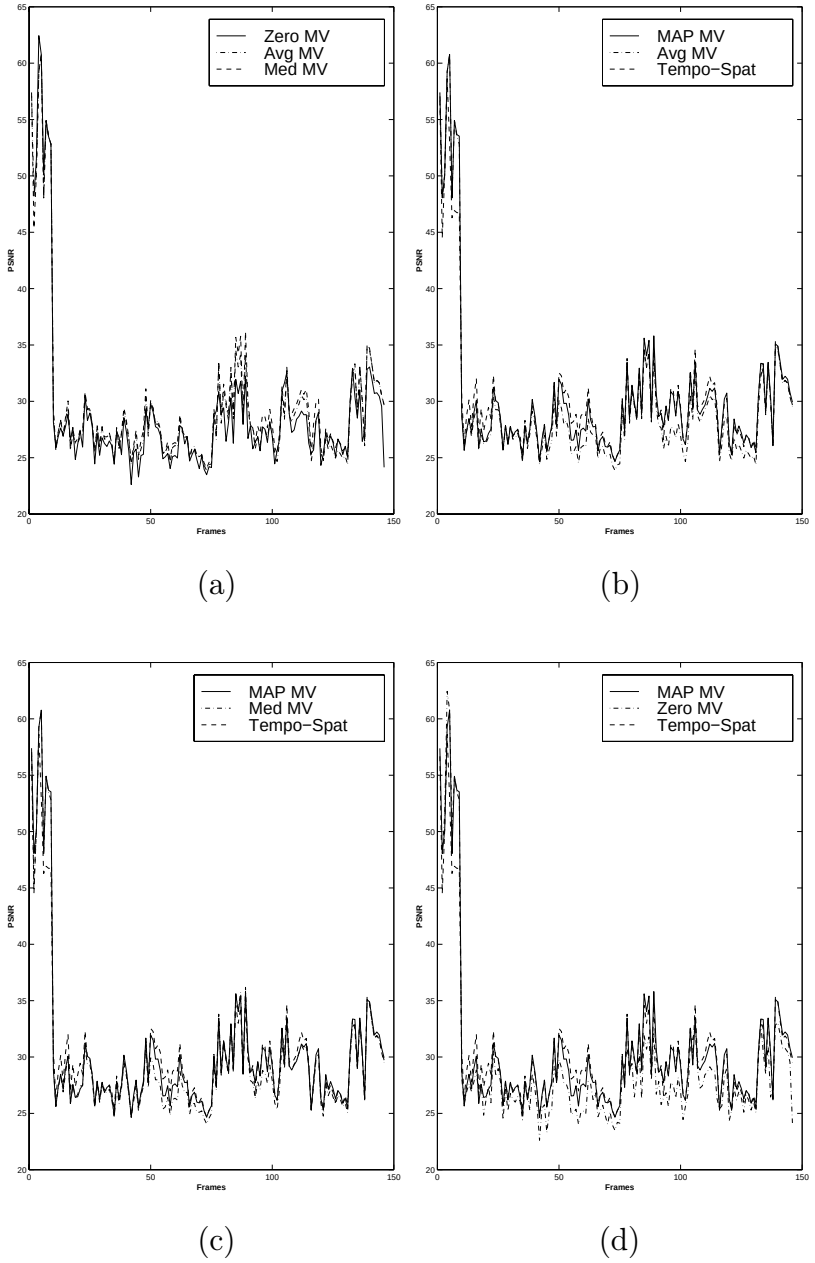


Fig. 7.22. Reconstruction PSNR values for the *footballl* sequence when 10 percent of the cells were dropped. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

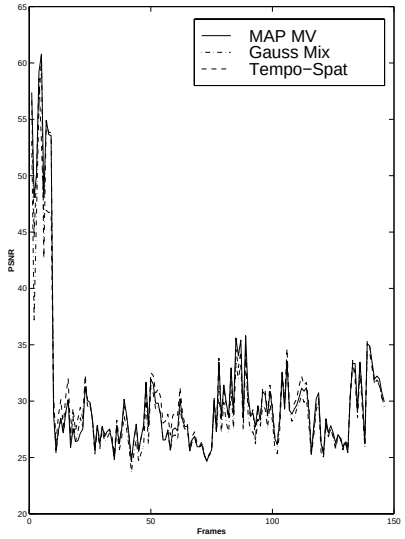


Fig. 7.23. (continuation of the previous figure) Reconstruction PSNR values for the *football* sequence when 10% of the cells were dropped: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

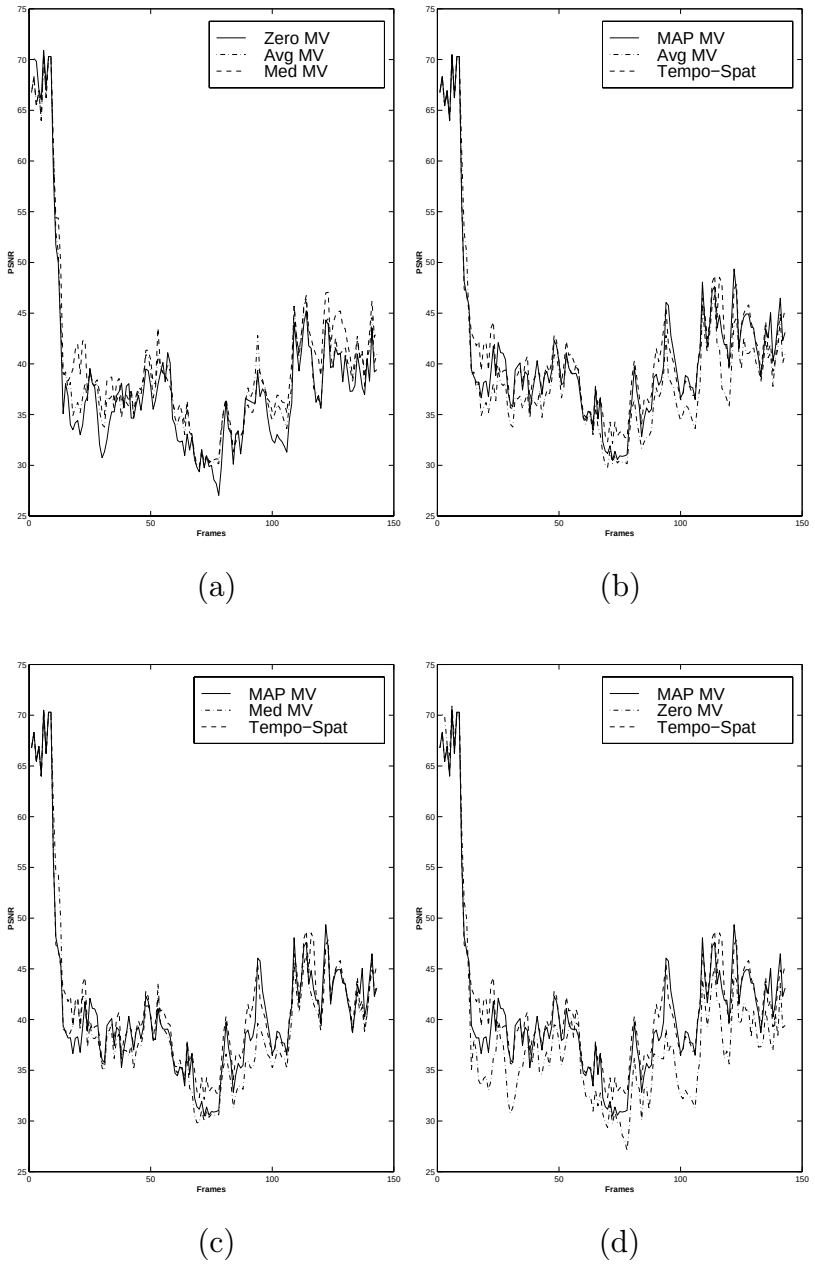


Fig. 7.24. Reconstruction PSNR values for the *hockey* sequence when 2 percent of the cells were dropped. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

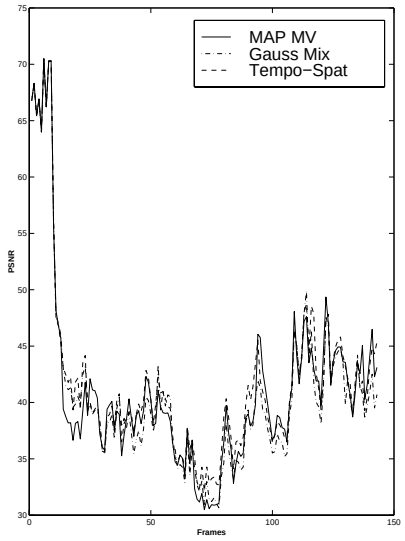


Fig. 7.25. (continuation of the previous figure) Reconstruction PSNR values for the *hockey* sequence when 2% of the cells were dropped: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

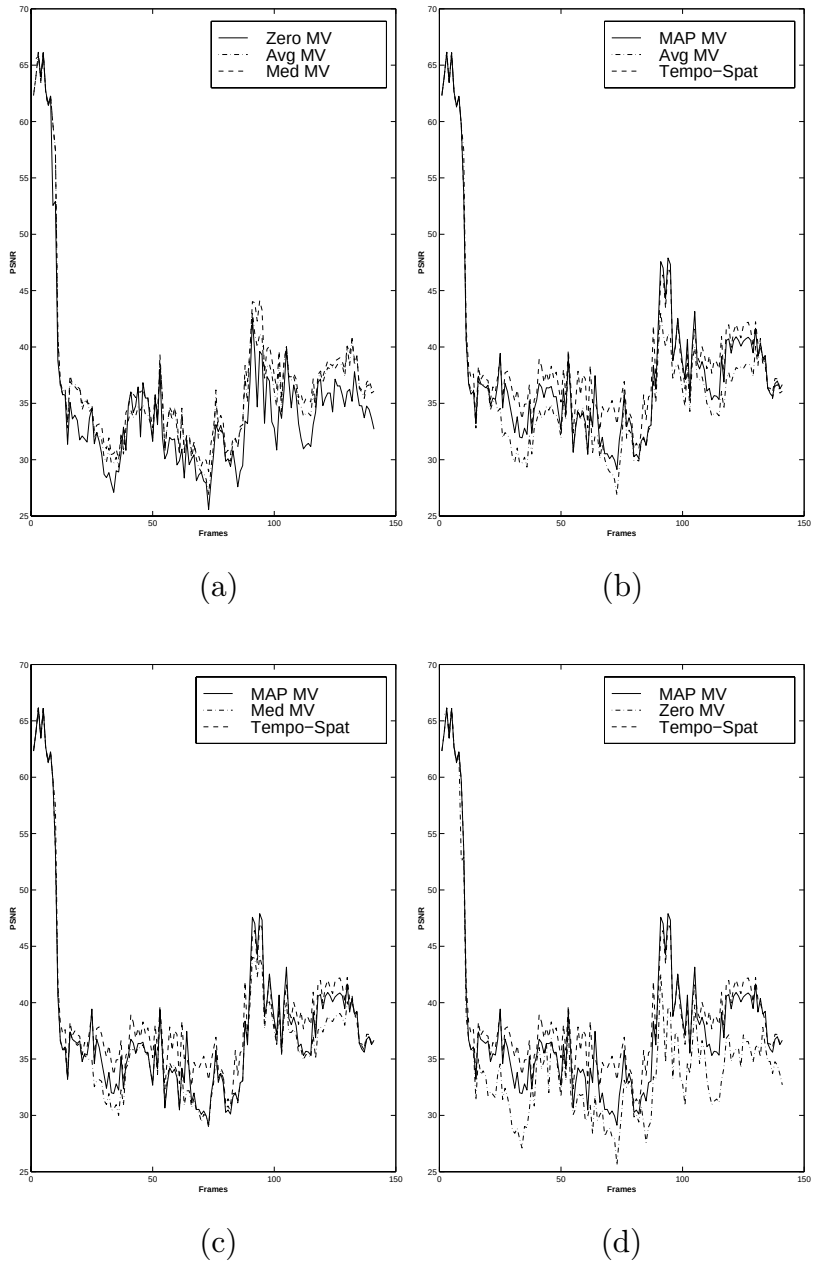


Fig. 7.26. Reconstruction PSNR values for the *hockey* sequence when 5 percent of the cells were dropped. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

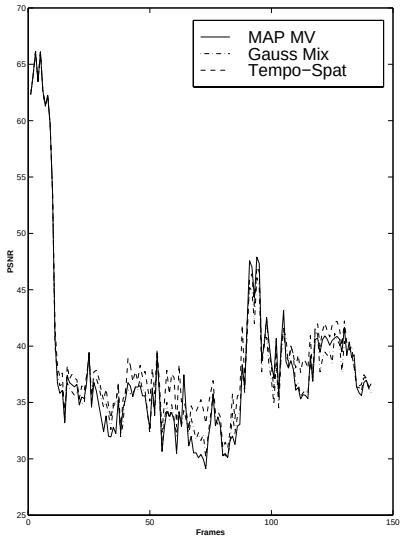


Fig. 7.27. (continuation of the previous figure) Reconstruction PSNR values for the *hockey* sequence when 5% of the cells were dropped: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

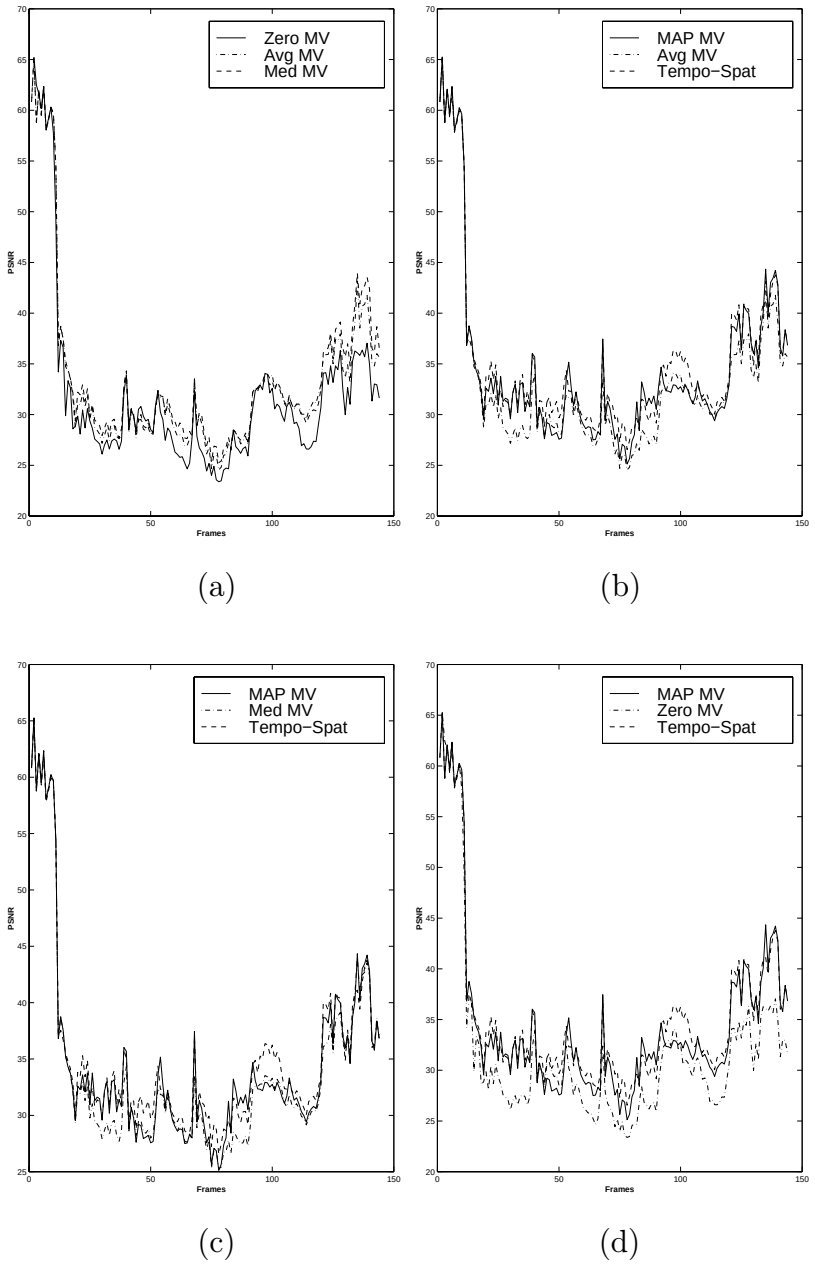


Fig. 7.28. Reconstruction PSNR values for the *hockey* sequence when 10 percent of the cells were dropped. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

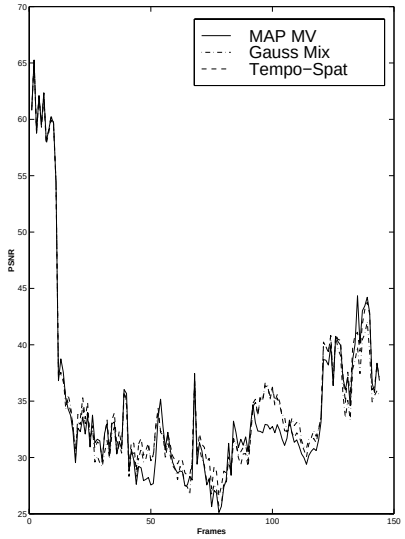


Fig. 7.29. (continuation of the previous figure) Reconstruction PSNR values for the *hockey* sequence when 10% of the cells were dropped: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

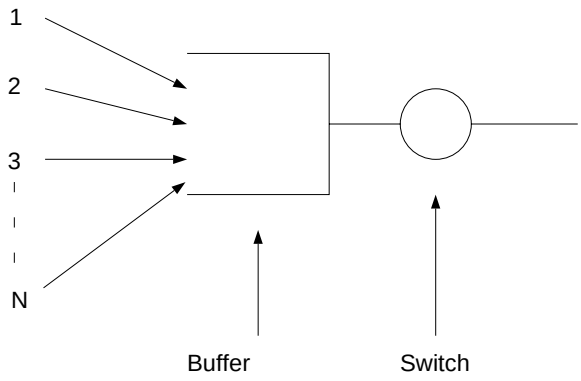
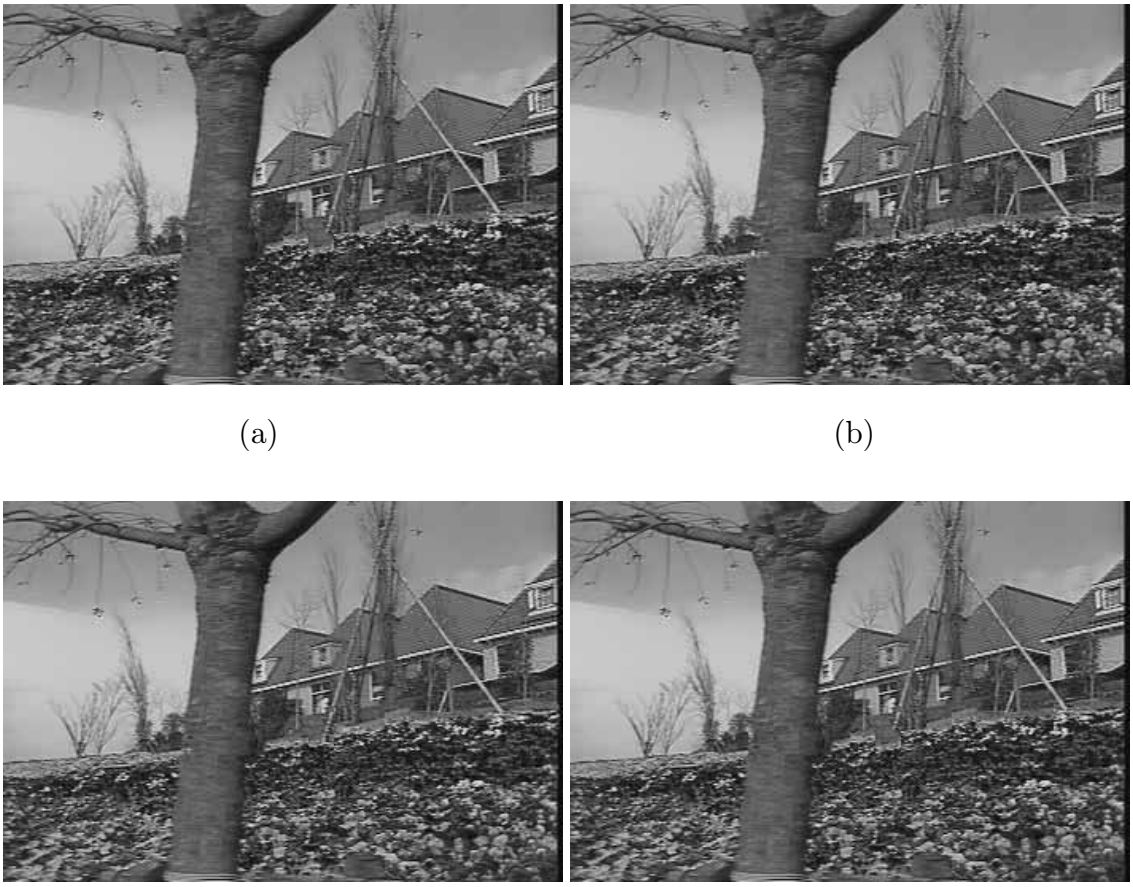


Fig. 7.30. Data streams arriving at a switching element

(a) (b)

(c) (d)

Fig. 7.31. (a): decoded frame from the *flowergarden* sequence, (b): frame is damaged due to 1% ATM cell loss, (c): the frame was restored by using temporal replacement, (d): the frame was reconstructed by finding the average of the neighboring motion vectors. The PSNR values are 32.36 dB and 32.36 dB respectively.



(a) (b)

(c) (d)

Fig. 7.32. (continuation of previous figure) (a): the frame was restored by finding the median of the neighboring motion vectors, (b): the frame was reconstructed by finding the MAP estimate of the missing motion vector, (c): the frame was restored by using the temporal-spatial approach, and (d): the frame was reconstructed using the Gaussian mixture model. The PSNR values are 32.36 dB, 32.89, 32.09, and 32.36 dB respectively.

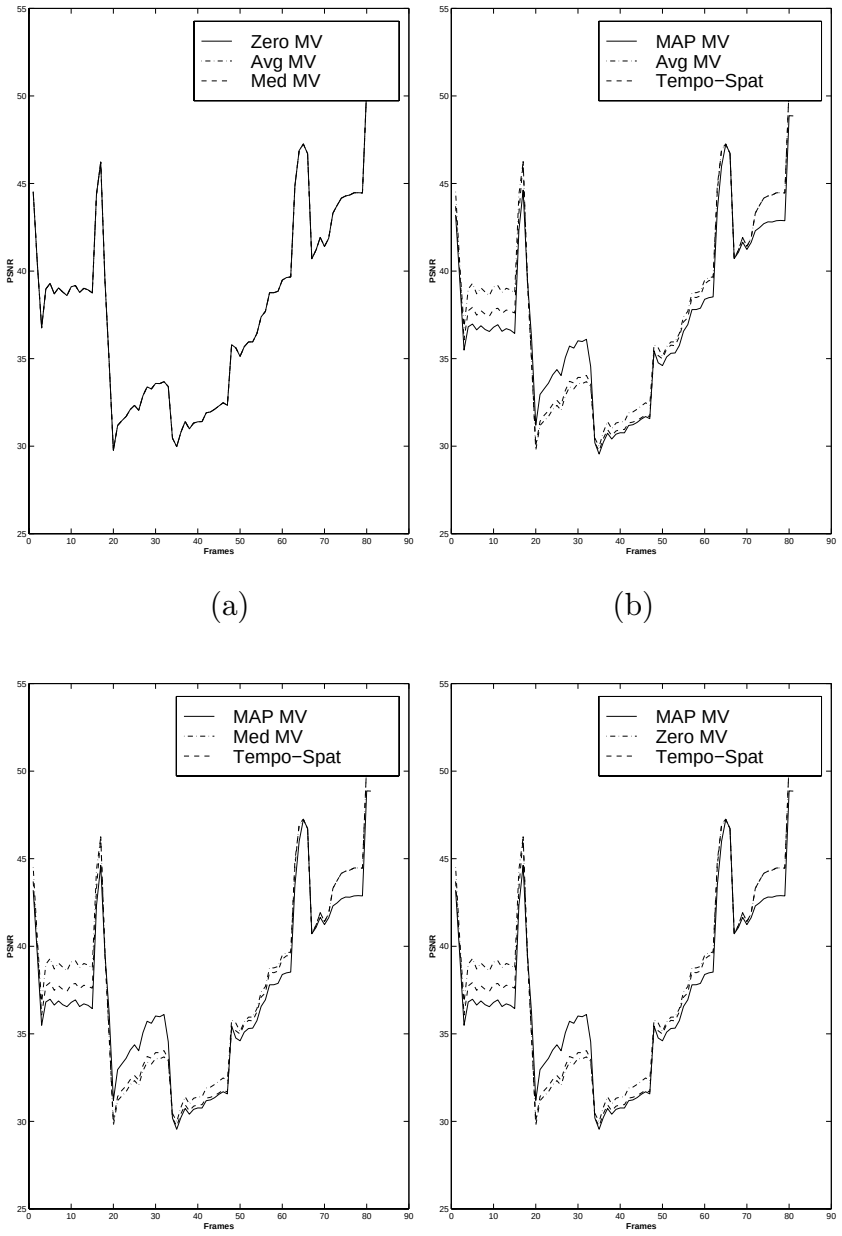


Fig. 7.33. Reconstruction PSNR values for the *flowergarden* sequence when 0.2% of the cells were dropped due to buffer overflow. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

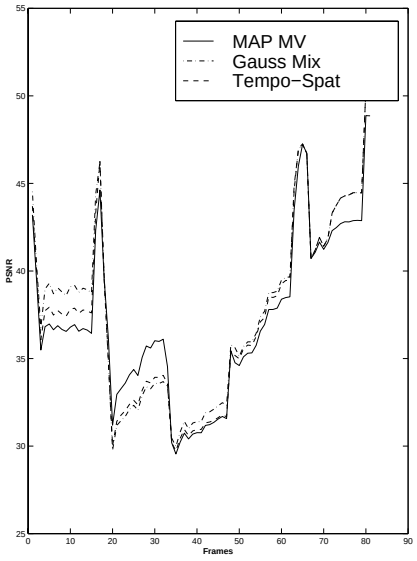


Fig. 7.34. (continuation of the previous figure) Reconstruction PSNR values for the *flowergarden* sequence when 0.2% of the cells were dropped due to buffer overflow: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

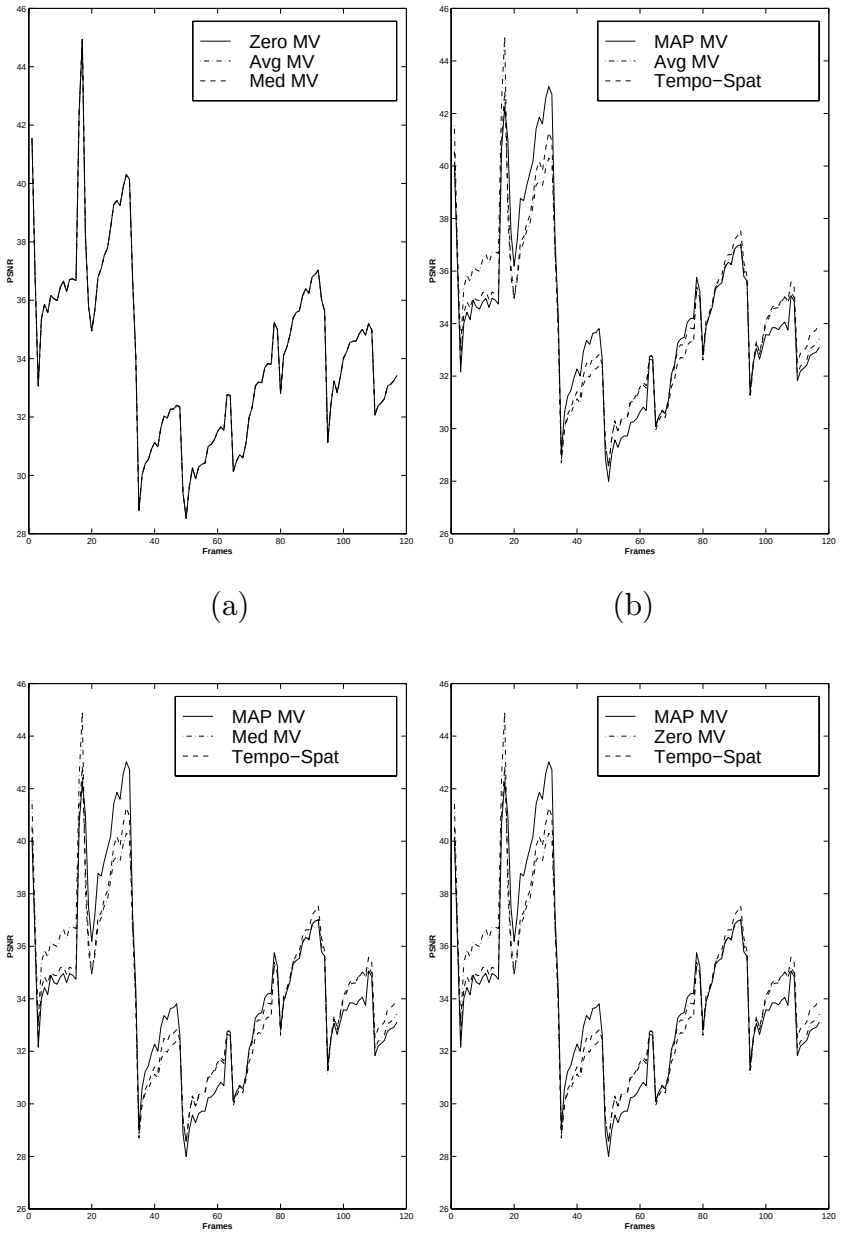


Fig. 7.35. Reconstruction PSNR values for the *flowergarden* sequence when 0.5% of the cells were dropped due to buffer overflow. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

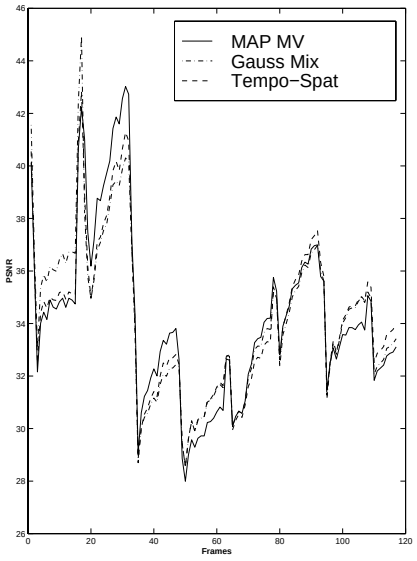


Fig. 7.36. (continuation of the previous figure) Reconstruction PSNR values for the *flowergarden* sequence when 0.5% of the cells were dropped due to buffer overflow: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

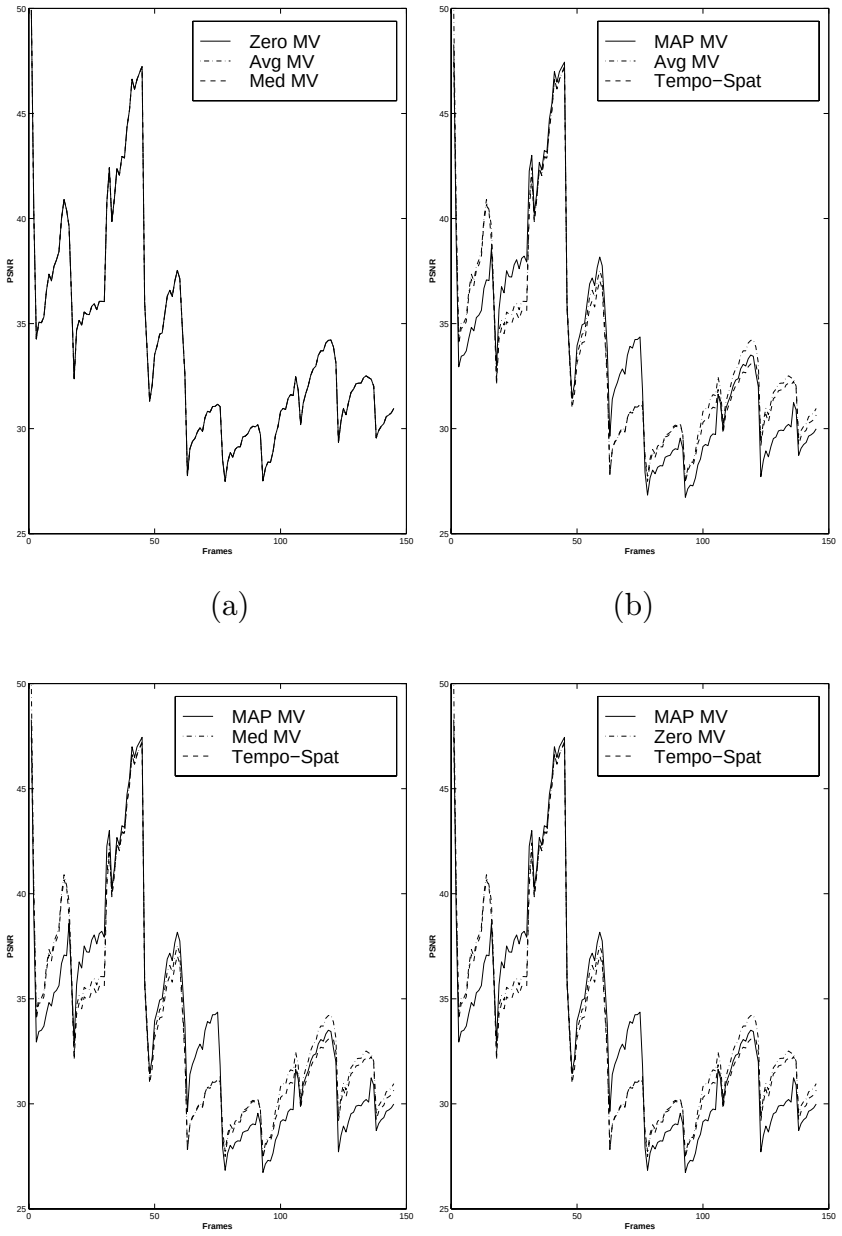


Fig. 7.37. Reconstruction PSNR values for the *flowergarden* sequence when 1% of the cells were dropped due to buffer overflow. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

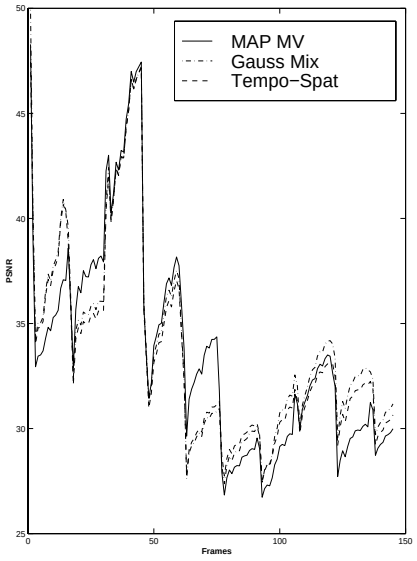


Fig. 7.38. (continuation of the previous figure) Reconstruction PSNR values for the *flowergarden* sequence when 1% of the cells were dropped due to buffer overflow: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

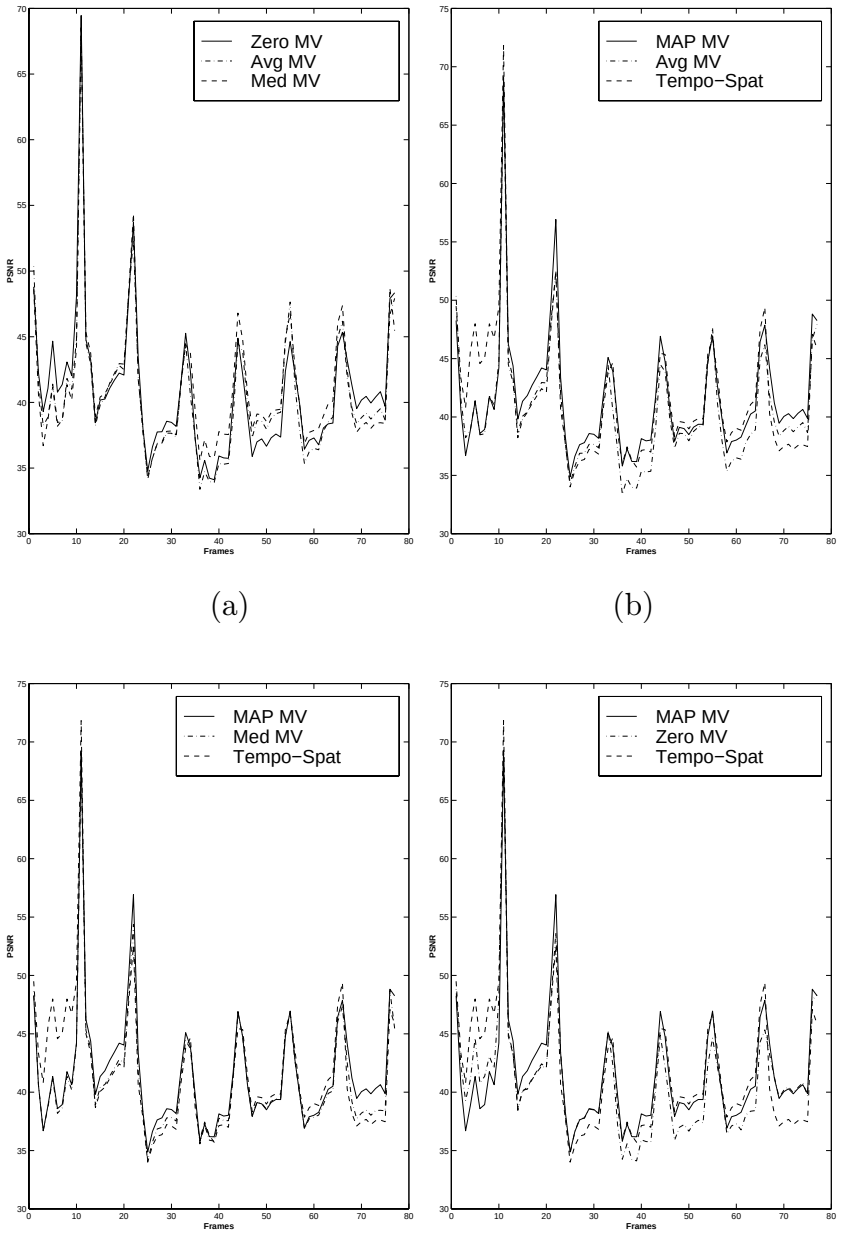


Fig. 7.39. Reconstruction PSNR values for the *football* sequence when 0.2% of the cells were dropped due to buffer overflow. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

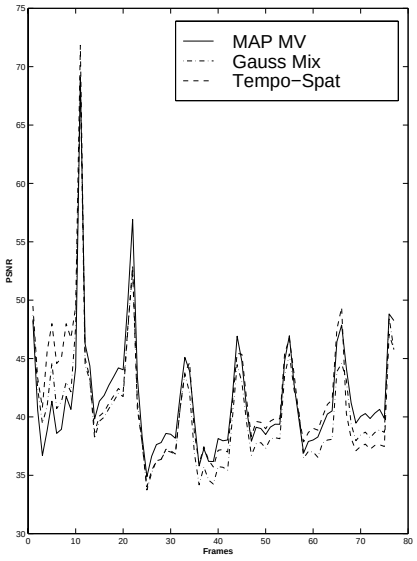


Fig. 7.40. (continuation of the previous figure) Reconstruction PSNR values for the *football* sequence when 0.2% of the cells were dropped due to buffer overflow: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

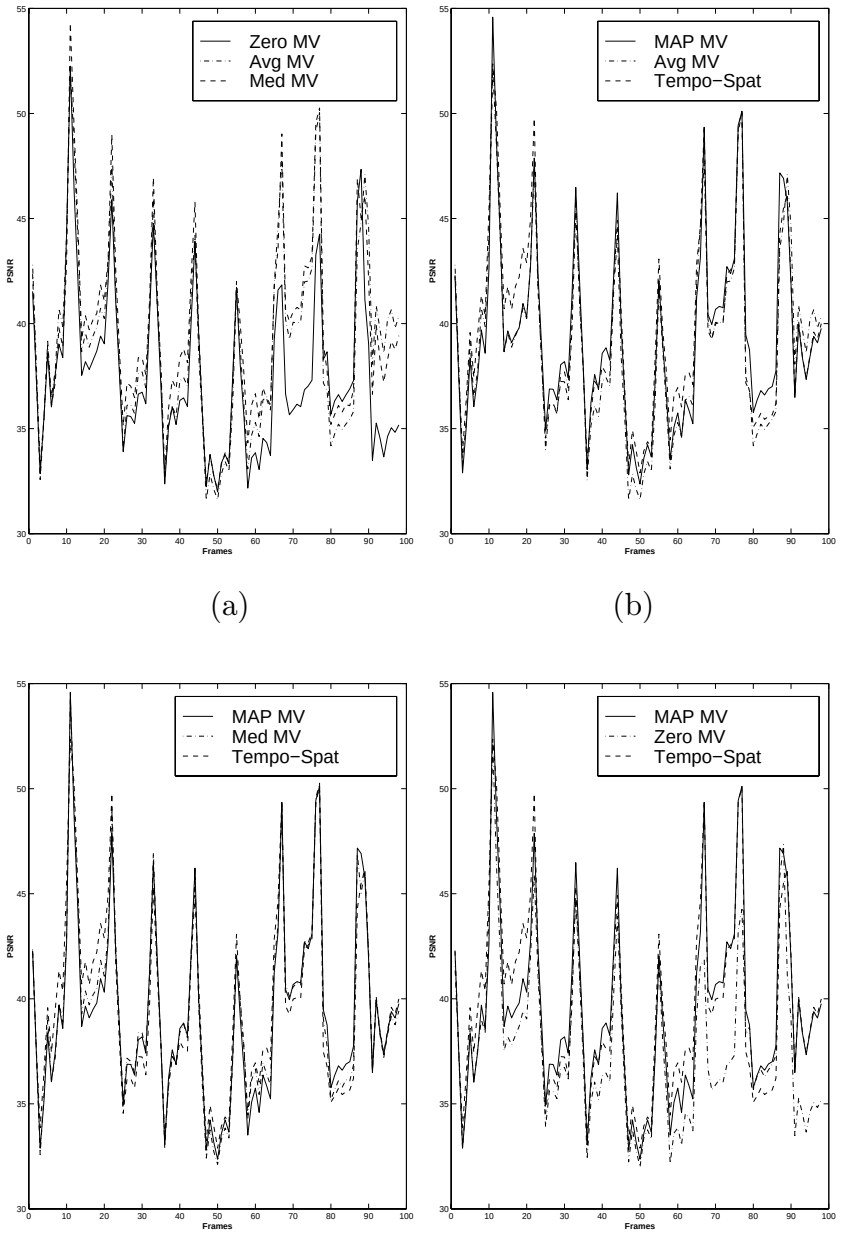


Fig. 7.41. Reconstruction PSNR values for the *football* sequence when 0.5% of the cells were dropped due to buffer overflow. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

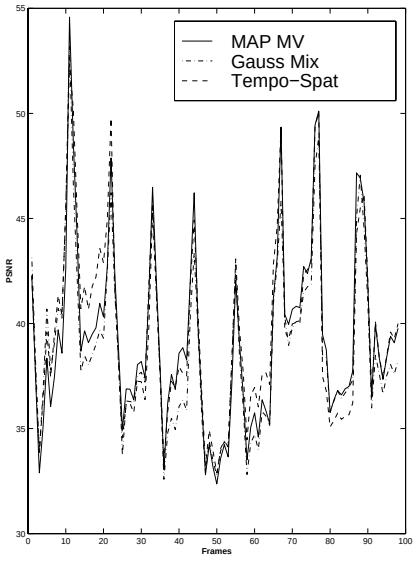


Fig. 7.42. (continuation of the previous figure) Reconstruction PSNR values for the *football* sequence when 0.5% of the cells were dropped due to buffer overflow: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

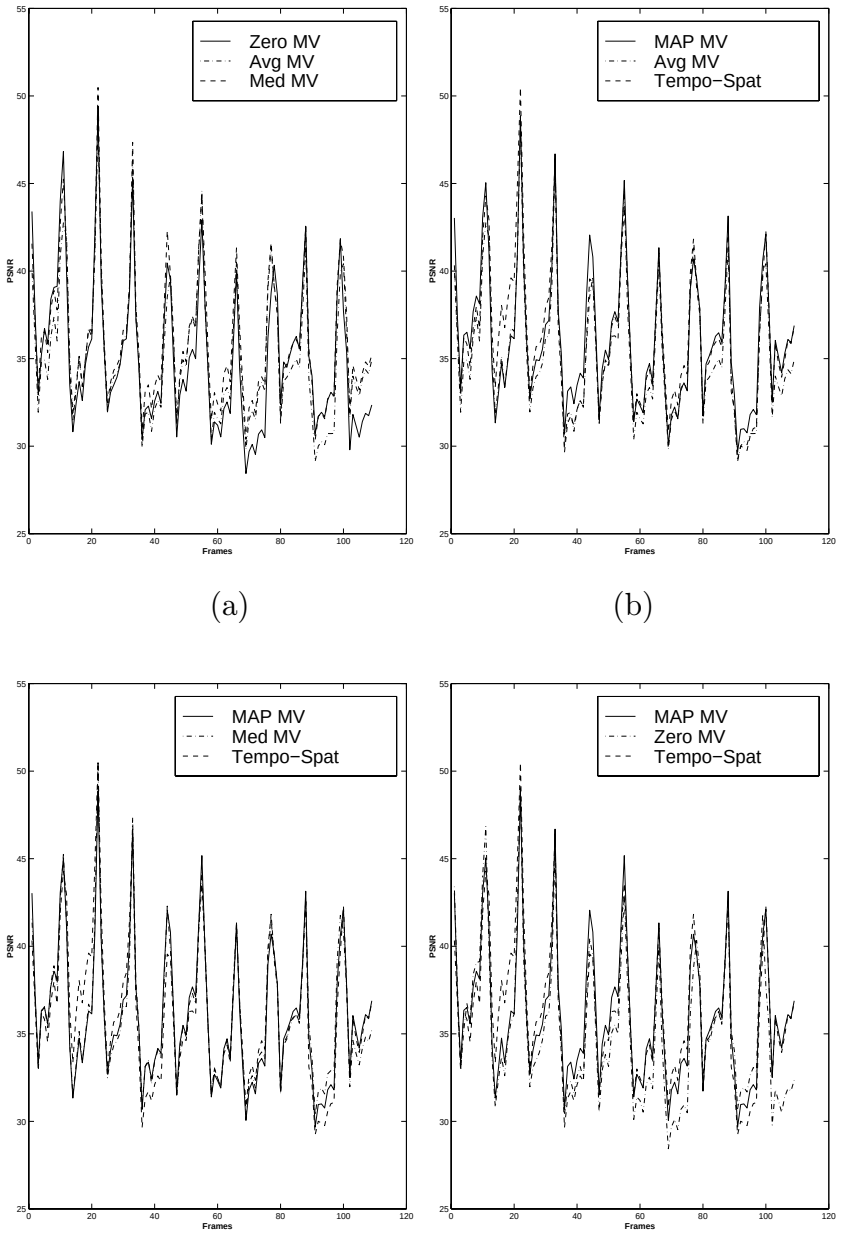


Fig. 7.43. Reconstruction PSNR values for the *football* sequence when 1% of the cells were dropped due to buffer overflow. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

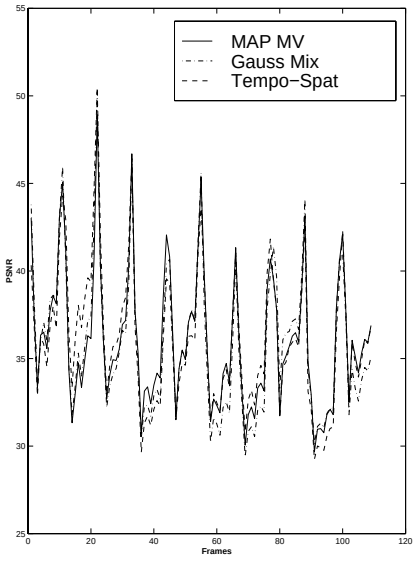


Fig. 7.44. (continuation of the previous figure) Reconstruction PSNR values for the *football* sequence when 1% of the cells were dropped due to buffer overflow: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

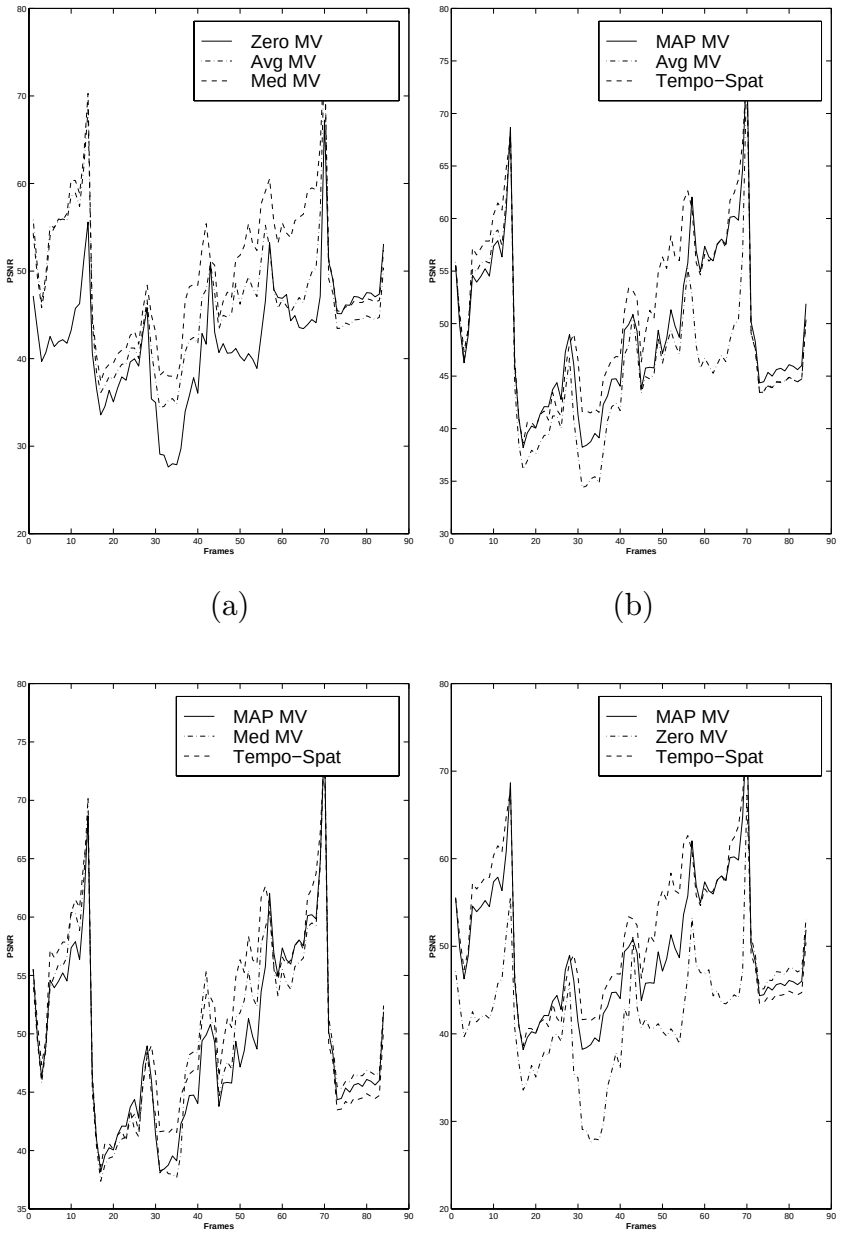


Fig. 7.45. Reconstruction PSNR values for the *hockey* sequence when 0.2% of the cells were dropped due to buffer overflow. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

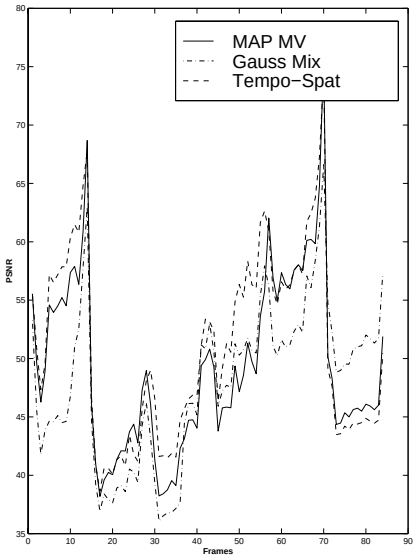


Fig. 7.46. (continuation of the previous figure) Reconstruction PSNR values for the *hockey* sequence when 0.2% of the cells were dropped due to buffer overflow: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

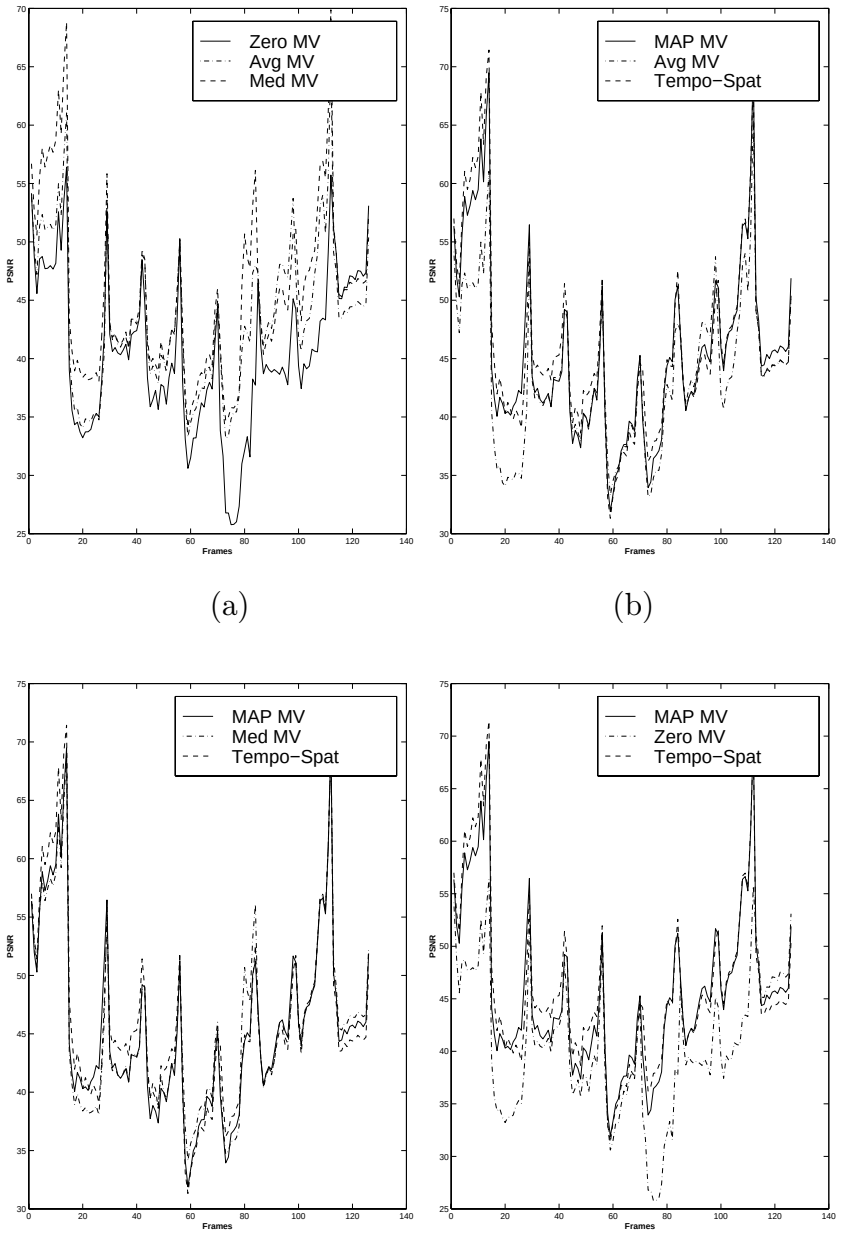


Fig. 7.47. Reconstruction PSNR values for the *hockey* sequence when 0.5% of the cells were dropped due to buffer overflow. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

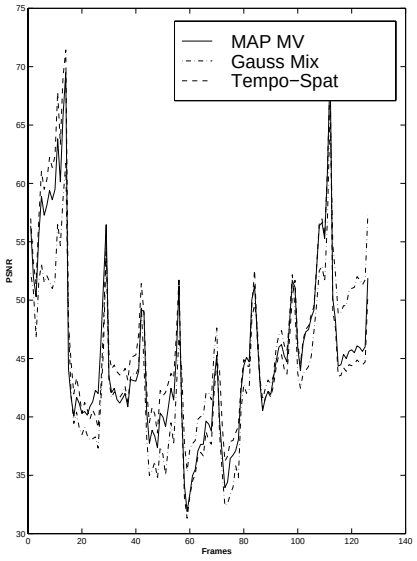


Fig. 7.48. (continuation of the previous figure) Reconstruction PSNR values for the *hockey* sequence when 0.5% of the cells were dropped due to buffer overflow: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

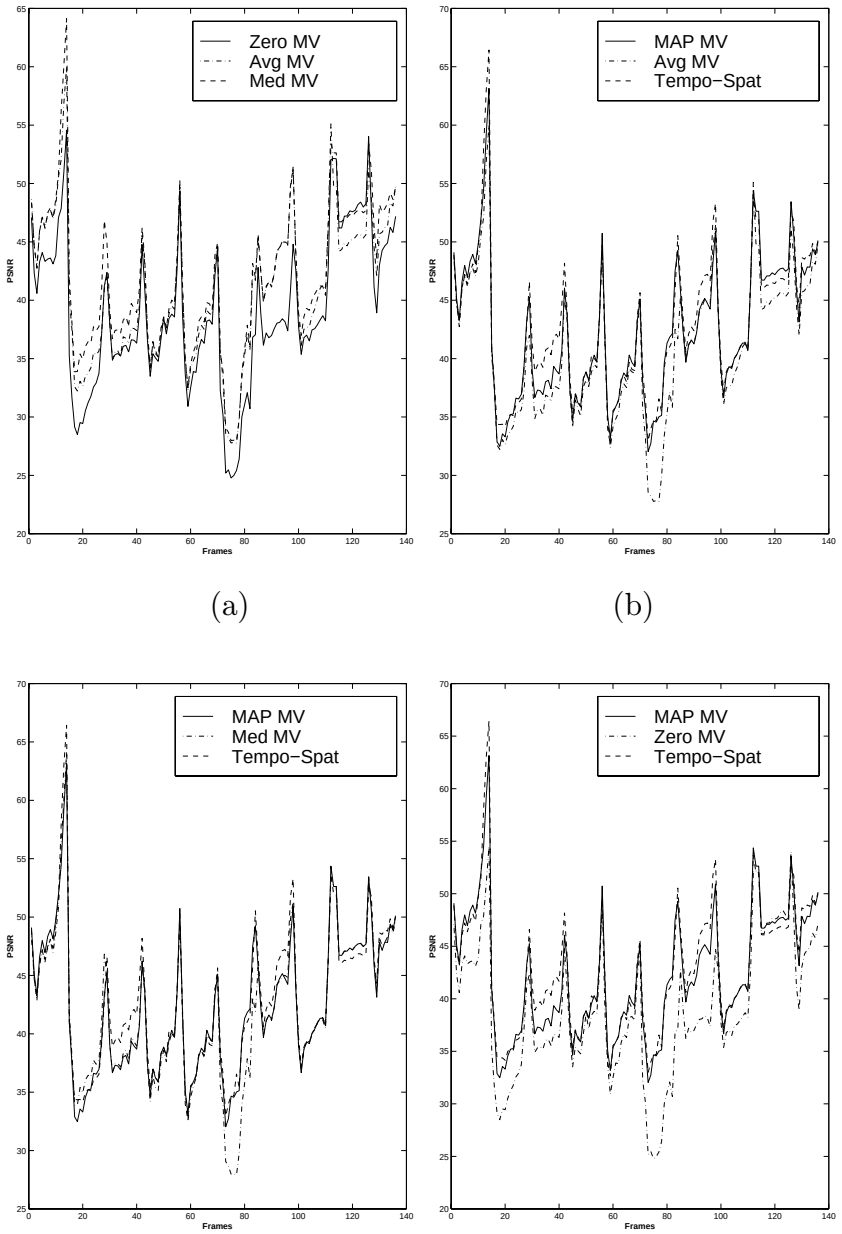


Fig. 7.49. Reconstruction PSNR values for the *hockey* sequence when 1% of the cells were dropped due to buffer overflow. (a): zero motion vector (temporal replacement), the average of the neighboring motion vectors, and the median of the neighboring motion vectors used. (b): the MAP estimate of the missing motion vector, the temporal spatial approach, and the average of the neighboring motion vectors are used. (c): the MAP estimate of the missing motion vector, the temporal spatial approach, and the median of the neighboring motion vectors are used. (d): the MAP estimate of the missing motion vector, the temporal spatial approach, and temporal replacement are used.

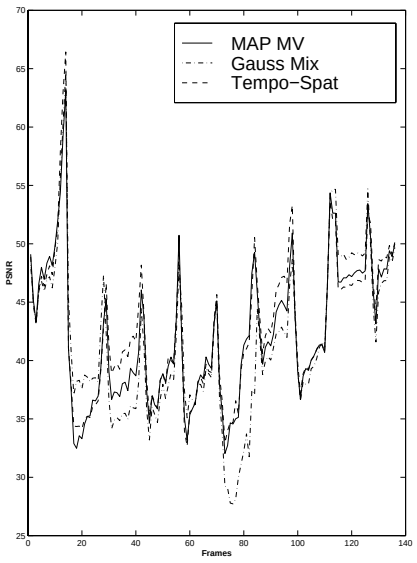


Fig. 7.50. (continuation of the previous figure) Reconstruction PSNR values for the *hockey* sequence when 1% of the cells were dropped due to buffer overflow: the MAP estimate of the missing motion vector, the temporal spatial approach, and Gaussian Mixture model are used.

8. EZW ERROR CONCEALMENT

8.1 Introduction

Of the many types of compressed video sequences that are attractive for transmission over broadband networks, are those that are *continuously scalable*. Continuous rate scalability provides the capability of arbitrarily selecting a specific data rate from within a range of data rates. Thus, a video server can couple the data rate of the video being delivered to the available network bandwidth.

A specific coding strategy known as *embedded rate scalable coding* [9, 88] is well suited for continuous rate scalable applications. In embedded coding, all the compressed data is embedded in a single bit stream and can be decoded at different data rates. A decoder then has the choice of decoding the compressed data from the beginning of the bit stream up to a point where a certain data rate requirement is achieved. A decompressed image at that data rate can then be reconstructed and the visual quality corresponding to this data rate achieved. Thus, to achieve best performance the bits that convey the most important information need to be embedded at the beginning of the compressed bit stream. In video compression, the presence of multiple images adds to the complexity of the coding strategy. In this case, the transmitter needs to selectively provide the decoder with portions of the bit stream corresponding to different frames or sections of frames of the video sequence, rather than sending the beginning portion of the bit stream to the decoder. These selected portions of the compressed data achieve the data rate requirement and can then be decoded by the decoder. This approach can be achieved if the position of the bits corresponding to each frame or each section of frames can be identified. Several embedded rate scalable video coding schemes have been proposed [88, 89, 90, 91, 92, 93].

8.2 Overview of Embedded Zerotree Wavelet Coding

In the embedded zerotree wavelet (EZW) coding strategy, developed by Shapiro [9], a wavelet/subband decomposition of the image is performed. The wavelet coefficients/pixels are then grouped into Spatial Orientation Trees [88]. The magnitude of each wavelet coefficient/pixel in a tree, starting with the root of the tree, is then compared to a particular threshold T. If the magnitudes of all the wavelet coefficients/pixels in the tree are smaller than T, the entire tree structure (that is the root and all its descendant nodes) is coded by one symbol, the zerotree symbol ZTR [9]. If however, there exit significant wavelet coefficients/pixels, then the tree root is coded as being significant or insignificant (the symbol IZ, isolated zero, being used in this case), if its magnitude is larger than or smaller than T, respectively. The descendant nodes are then each examined in turn to determine whether each is the root of a possible subzerotree structure, or not. This process is carried out such that all the nodes in all the trees are examined for possible subzerotree structures. The significant wavelet coefficients/pixels in a tree are coded by one of two symbols, POS or NEG, depending on whether their actual values are positive or negative, respectively. The process of classifying the pixels as being ZTR, IZ, POS, or NEG is referred to as the dominant pass in [9]. This is then followed by the subordinate pass in which the significant wavelet coefficients/pixels in the image are refined by determining whether their magnitudes lie within the intervals [T,3T/2) and [3T/2,2T). Those wavelet coefficients/pixels whose magnitudes lie in the interval [T,3T/2) are represented by a 0 (LOW), whereas those with magnitudes lying in the interval [3T/2,2T) are represented by a 1 (HIGH). Subsequent to the completion of both the dominant and subordinate passes, the threshold value T is reduced by a factor of 2, and the entire process repeated. This coding strategy, consisting of the dominant and subordinate passes followed by the reduction in the threshold value, is iterated until a target bit rate is achieved.

The tree structure used in [9] is shown in Figure 8.4. The root node of each tree is located at the highest level of the decomposition pyramid, and all its descendants (indicated by the arrows) are located in different spatial frequency bands at the same pyramid level, or clustered in groups of 2×2 at lower levels of the decomposition pyramid.

An EZW decoder reconstructs the image by progressively updating the values of each wavelet coefficient/pixel in a tree as it receives the data. The decoder's decisions are always synchronized to those of the encoder.

8.3 Problems with EZW

Although EZW is very efficient and effective in reducing the spatial redundancies in an image, yet it is very susceptible to transmission errors. The misinterpretation of a symbol by the decoder, as a result of transmission errors, is sufficient enough to result in loss of synchronization between the encoder and decoder. Consider the case of a tree structure being coded by the ZTR symbol, yet the data that the decoder receives indicates otherwise. In this case the decoder will be expecting extra data in order to reconstruct the wavelet coefficients/pixels belonging to the tree. Since there is originally no data for those wavelet coefficients/pixels, the decoder will then commence to decode the data belonging to other trees and assign it to the current tree. In fact the misinterpretation of only one symbol is sufficient to derail the entire decoding process. The effect of incorrectly decoding a ZTR symbol as a POS symbol is observed in Figure 8.1b.

Our experiments have shown that of the four symbols used to encode an image, the most important and which must be protected are the IZ (isolated zero) and ZTR (zero tree root) symbols. They also however, comprise on average about 60 percent and 25 percent, respectively, of all the symbols used to code an image. Our experiments have also shown that the misinterpretation of the significant wavelet coefficients/pixels do not impact the subjective quality of the reconstructed image as severely as an error in decoding the IZ and ZTR symbols. Thus, the objective of any strategy used

to pack the coded bitstream into network packets would be to protect the IZ and ZTR symbols. Thus, unequal error protection whereby sections of the bitstream are afforded higher protection than other sections is necessary. The packetization strategy should also permit the loss of the significant wavelet coefficients/pixels without any loss of synchronization between the encoder and decoder. One naive approach would be to pack all the ZTR symbols into high priority packets, in particular ATM cells, and interpret any missing symbol as being an IZ symbol. However, it has been observed that at least 40 percent of all the cells used to transmit the data have to be designated as being high priority cells.

Performing error concealment for sequences encoded via EZW is to be contrasted with that of concealing errors in MPEG encoded video sequences [56, 55]. Although MPEG sequences are not rate scalable, yet they are more robust to transmission errors/packet loss. There is enough redundant spatial and temporal data, that can be exploited to reduce the effect of transmission errors/packet loss.

8.4 Approaches to Error Concealment in EZW

We present here a brief overview of the different approaches used by other authors [94, 2, 95, 96] to conceal the effect of transmission errors in EZW encoded video streams.

In [94], all the zerotrees are grouped together into groups of N trees that are interleaved together to form M separate bitstreams. The M bitstreams are then transmitted separately. If an error is detected in a bitstream the decoder stops decoding it and commences decoding another bitstream. A similar approach was adopted in [96], wherein the N interleaved trees are coded such that the resulting bitstream is exactly 48 bytes, an ATM cell user payload. If the size of the resulting bitstream is longer than 48 bytes, the trees are pruned. If on the other hand, the length of the generated bitstream is less than 48 bytes, the trees are coded at a higher bit rate than the target bit rate. The root nodes of the trees chosen for interleaving are not spatially adjacent. This has the advantage that the loss of one cell will not affect the

decoding process. Furthermore, the missing wavelet coefficients/pixels can then be interpolated from their neighboring wavelet coefficients/pixels or set to be zero. The disadvantage is the loss of rate scalability. Image reconstruction by the decoder is no longer progressive.

In [95], the authors use Error-Resilient Entropy Coding (EREC) [97] to map blocks of variable length data into blocks of fixed length, called frames. For EREC to work, the coded data to be sent has to satisfy the prefix condition, that is no code is a prefix of another code. At the receiving end, the decoder exploits the fact that the codes are prefix codes to detect the presence of errors in the bitstream and circumvent their effect on the quality of the decoded image.

A variation of the embedded zerotree wavelet codec developed by Said and Pearlman [88] was proposed in [2]. The new coder produces two bitstreams, denoted as the MAP and QUAN bitstreams. The QUAN bitstream consists of fixed length codes representing the quantizations of the significant wavelet coefficients/pixels. The coder constructs the MAP bitstream by producing a 1 whenever a coefficient/pixel or one of its descendants is significant relative to the threshold value at the current dominant pass, and a 0 whenever a coefficient/pixel and all its descendants are insignificant relative to the threshold value. Unequal error protection [3] is then employed to provide high protection to the MAP bitstream, and moderate and low protection to the QUAN bitstream. Although not as efficient as the Said and Pearlman algorithm, this technique is more robust in the presence of transmission errors.

It is to be emphasized that with the exception of [96], all authors test the robustness of their techniques by simulating binary symmetric channels, in which the data in the bitstream is corrupted. This is to be contrasted with the problem at hand in which entire blocks of data from the bitstream, 384 bits in particular, are lost.

Currently robustness measures for the image compression techniques being considered for JPEG2000 are being investigated. The compression technique is based on the Wavelet Trellis Coded Quantization (WTCQ) algorithm [98], which utilizes

the wavelet transform and trellis coded quantization to quantize the wavelet coefficients [99]. The quantized coefficients are grouped into structures that span across subbands [98]. Each structure is entirely binary coded by an arithmetic encoder separately from the other structures. The compressed data belonging to the structures are separated by extra added resynchronization fields. These fields are used by the decoder to determine corrupted structures. The wavelet coefficients belonging to a corrupted structure are set to zero.

Alternative protection of coded image data is described in [100] through the use Multiple Description Coding. The source encoder produces a number of correlated output sequences that are to be sent over separate channels with varying transmission reliabilities. Data corrupted in any particular group of channels is restored by means of the remaining uncorrupted data. The separate sequences are then used by the decoder to reconstruct the image. It is to be noted that Multiple Description Coding is independent of the transform used in the image compression paradigm.

8.5 Alternative Approach

Our approach to error concealment in EZW compressed images transmitted across data networks, in particular ATM networks, is based upon the use of unequal error protection, and data interleaving. Unequal error protection is achieved through the use of 2 different convolutional codes, namely (3,2,3), and (4,3,2) [70]. This is in contrast to the technique described in [2], which relies upon the use of Rate Compatible Punctured Convolutional(RCPC) codes [3]. The disadvantage of using RCPC codes is that the effect of puncturing, used to achieve low rates, cannot be reversed by a decoder using Hard Decisions. The advantage, however, is a decrease in the encoder and decoder complexity. The (3,2,3) and (4,3,2) encoders are employed to provide two different levels of protection for the compressed image data. It was observed that the incurred overhead was 40%. It is to be noted that the other codes besides convolutional codes could have been utilized to achieve unequal error protection [101, 102, 103].

The compressed image data is segmented into blocks of size 1152 bytes (the size of 24 ATM cells). This is done to reduce decoding delay and preserve rate scalability, while providing less overhead than that incurred if smaller block sizes are used. The segmented data is subsequently encoded by one of the 2 convolutional encoders, the particular encoder being chosen according to the priority of the data being sent. Each coded block of data is then divided into groups of 384 bytes (the size of 8 ATM cells). The data in each group is interleaved to disperse the effect of packet loss over the entire coded block [59, 104]. After interleaving the data is sent across a data network. This was modeled as a queue with a fixed service rate μ and a finite input buffer. Data packets arriving to the buffer were queued for service if the buffer capacity had not been exceeded, otherwise they were discarded. The arrival rate, λ, of incoming packets depended on the number packets per coded image. It is assumed that initially the input buffer is 95% full. This is to prevent the buffer of the simulated node from discarding packets carrying data from only the latter part of the coded data stream. Figure 8.5 below, depicts the entire scheme.

At the receiving end, the entire process is reversed. Arriving blocks of data are de-interleaved, decoded by a Viterbi decoder [70, 71], de-segmented, and decoded by an EZW decoder. It is assumed that the receiver knows exactly which packets were discarded by the network, and replaces them by packets with a payload of zeros. In addition it is also assumed that the Viterbi decoder knows the exact coding pattern used by the encoder, that is which data blocks were coded by the different encoders. When computing the Maximum Likelihood (ML) path through the trellis, it is assumed that bit errors are independent of future and past bit errors, and occur with probability p [71]. This assumption does not hold for bits belonging to the same packet, however, yet it is used for simplicity. Furthermore, only hard decisions are made by the decoder.

8.6 Results

To test the performance of our scheme two images were each coded by an EZW encoder at a data rate of 2 bits per pixel. They were subsequently encoded by means of the convolutional encoders, interleaved, and subjected to 0.5%, 1%, 2%, and 5% ATM cell loss rates. At the receiver, the data was de-interleaved, Hard decoded by the Viterbi algorithm, and then decoded by an EZW decoder at data rates of 2, 1, 0.5, 0.2 bits per pixel, respectively.

Figures 8.2 (a) and 8.3 (a) depict the original *girls* and *airport* images respectively. The corresponding uncorrupted images decoded at a data rate of 2 bits per pixel (bpp) are shown in Figures 8.2 (b) and 8.3 (b), respectively. The outcome of decoding each image at a data rate of 1.5 bit per pixel after having suffered a 5% ATM cell loss is given in 8.2 (c) and 8.3 (c) respectively. Similarly, the outcome of decoding each image at a data rate of 0.5 bit per pixel after having suffered a 5% ATM cell loss is given in 8.2 (d) and 8.3 (d) respectively. For comparison purposes, the uncorrupted images decoded at data rates of 1.5 and 0.5 bits per pixel are provided in Figures 8.2 (e) and 8.3 (e), and Figures 8.2 (f) and 8.3 (f) respectively.

In Tables 8.1 and 8.2, below, the Peak Signal to Noise Ratio (PSNR) in dB, for the various reconstructed images at the different data rates and ATM cell loss rates are given. It is to be noted that the comparisons are made between the corrupted and uncorrupted decoded images, at the different data rates.

A disadvantage of the technique discussed above is the 40% overhead incurred. Such an overhead will further aggravate the network, resulting in excessive packet loss. In Chapter 9 we discuss one way for reducing this high overhead.

Table 8.1 PSNR values in dB for different data rates and cell loss rates for *girls*.

Data Rate	ATM Cell Loss rate			
	5%	2%	1%	0.5%
2 bpp	33.76	37.40	40.27	43.83
1.5 bpp	34.20	38.482	42.62	∞
1.0 bpp	35.19	42.21	∞	∞
0.5 bpp	39.82	∞	∞	∞
0.2 bpp	∞	∞	∞	∞

Table 8.2 PSNR values in dB for different data rates and cell loss rates for *airport*.

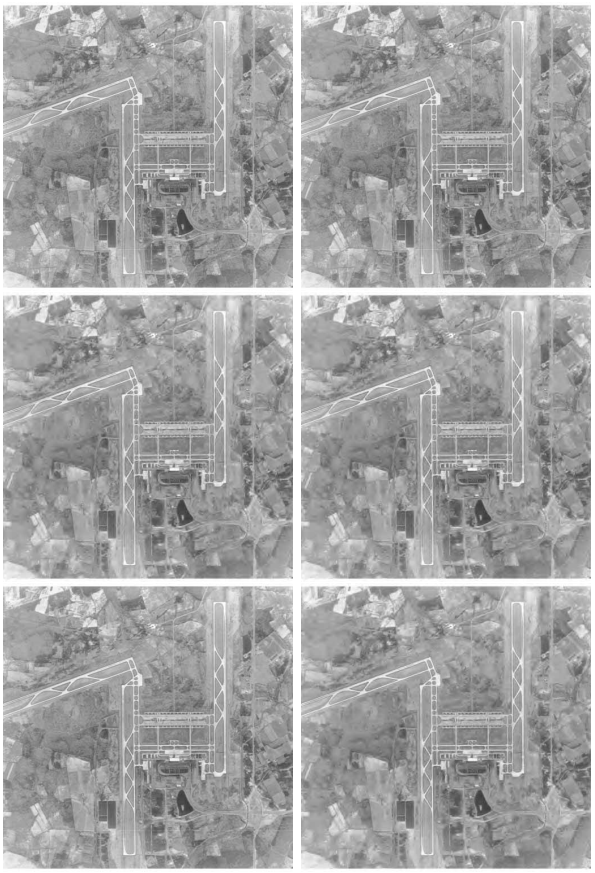
Data Rate	ATM Cell Loss rate			
	5%	2%	1%	0.5%
2 bpp	30.68	33.55	35.85	38.76
1.5 bpp	31.41	35.15	39.01	∞
1.0 bpp	31.96	37.14	∞	∞
0.5 bpp	39.20	∞	∞	∞
0.2 bpp	∞	∞	∞	∞

(a) (b)

Fig. 8.1. (a) Original image, (b) damaged image due to the misinterpretation of a ZTR symbol as a POS symbol.

(a) (b)

Fig. 8.2. (a) Original *girls* image, (b) uncorrupted image decoded at 2 bpps, (c) image decoded at 1.5 bpps after 5% of the ATM cells were lost, (d) image decoded at 0.5 bpps after 5% of the ATM cells were lost, (e) uncorrupted image decoded at 1.5 bpps, (f) uncorrupted image decoded at 0.5 bpps.



(a) (b)

Fig. 8.3. (a) Original *airport* image, (b) uncorrupted image decoded at 2 bpps, (c) image decoded at 1.5 bpps after 5% of the ATM cells were lost, (d) image decoded at 0.5 bpps after 5% of the ATM cells were lost, (e) uncorrupted image decoded at 1.5 bpps, (f) uncorrupted image decoded at 0.5 bpps.

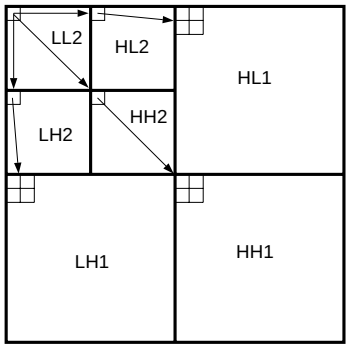


Fig. 8.4. Spatial orientation tree used by Shapiro.

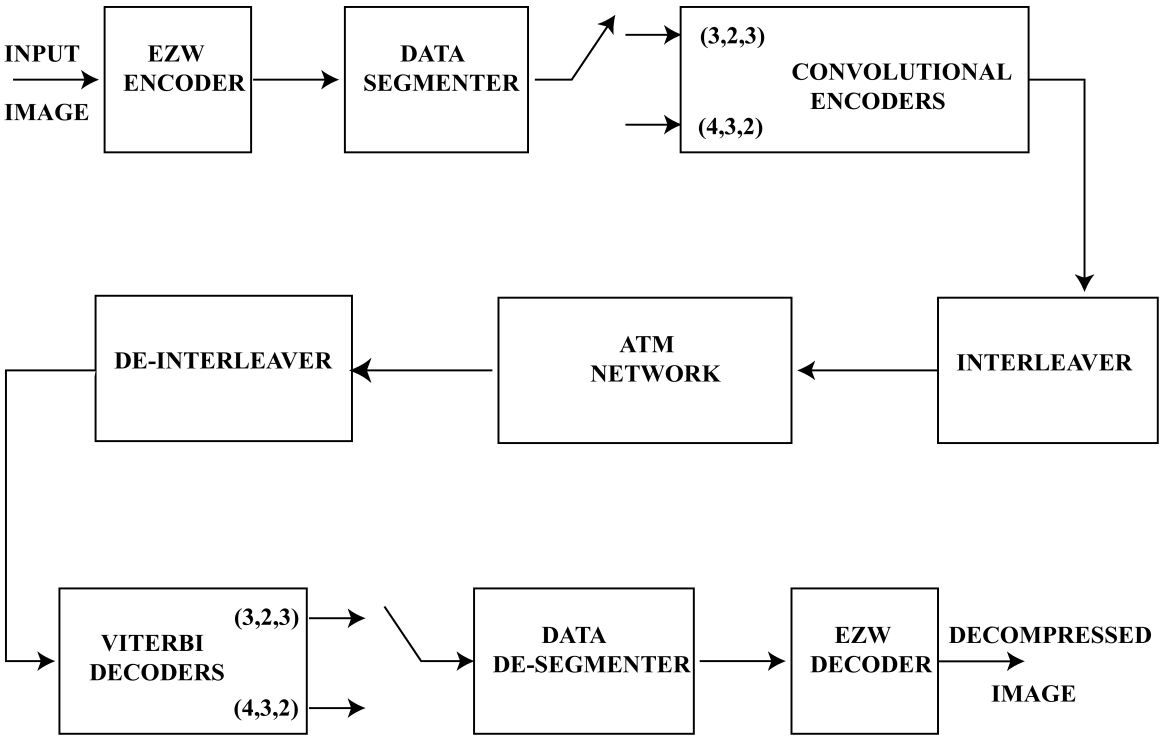


Fig. 8.5. Block diagram of entire system.

9. CONCLUSION AND FUTURE RESEARCH

In this thesis we have considered the following:

- How to pack MPEG-1 System Layer and MPEG-2 Program Layer Streams into network packets.
- Error concealment algorithms for reconstructing missing macroblocks and motion vectors.
- How to make Embedded Zerotree Wavelet Codecs resilient to network packet loss.

We proposed a new scheme for packing network packets with MPEG-1 System Layer Streams and MPEG-2 Program Layer Streams in Chapter 6. The objective was to minimize the impact of packet loss on the decoding process without incurring a high overhead, while protecting the least amount of information possible. This was achieved and it was observed that the overhead incurred due to the packing scheme did not significantly impact the packet loss process.

A general issue that also needs to be considered is the impact that protecting coded data has on the packet loss process. For instance, if all the motion vectors in a video sequence were protected, the reconstructed images at the receiving end would be of higher quality than if the motion vector data had not been protected. Protecting motion vector data can be achieved either by packing the data into High Priority packets, or by using Forward Error Correction (FEC) schemes. Using FEC schemes is advantageous however, since it requires the use of a smaller number of High Priority packets. This is important since in the event of network congestion, the network switching elements will not be forced to drop any High Priority packets.

It is however unknown if protecting motion vector data by using FEC schemes will further contribute to network congestion or not. This is one topic for future research.

In Chapter 7 we described spatial, temporal, and temporal-spatial techniques for error concealment. Our spatial techniques were based on a first order Huber Markov Random Field model. We also described a suboptimal spatial technique based on median filtering that can be implemented in real-time. While the advantage of using a first order model lies in the fact that it leads to a fast suboptimal error concealment algorithm, yet it is constrained to using boundary pixel values. Better reconstruction of macroblock data would be achieved if higher order models are utilized. However, such higher order models should not be too complex to be of practical use in real-time error concealment. Thus, the development of simple higher order models for spatial error concealment need to be investigated.

When comparing the various motion vector estimation techniques we observed that using the Temporal-Spatial approach was advantageous in the restoration of sequences in which both intracoded and intercoded frames were damaged. This is attributed to the fact that the Temporal-Spatial approach initially classifies the neighboring motion vectors into 9 nine classes, implicitly delineating the discontinuities in the motion field. This classification is then aided by the use of spatial data in determining an estimate for the missing motion vector. After having classified each neighboring motion vectors into its respective class, each class is assigned a cost, and the classes with the minimum cost functions are selected. The MAP estimate of the motion vector in a minimum cost class is obtained. Of all the possible candidates, the motion vector that results in a reconstruction with "best" matching boundaries is chosen. Rather than finding the MAP estimate for every class of motion vectors that has minimal cost, a search can be carried out for the motion vector belonging to that minimal cost class that matches the boundaries. In this case the search region need not be bounded by the motion vectors belonging to that class, but can exceed it. Although more computationally intensive than our Temporal-Spatial approach, this

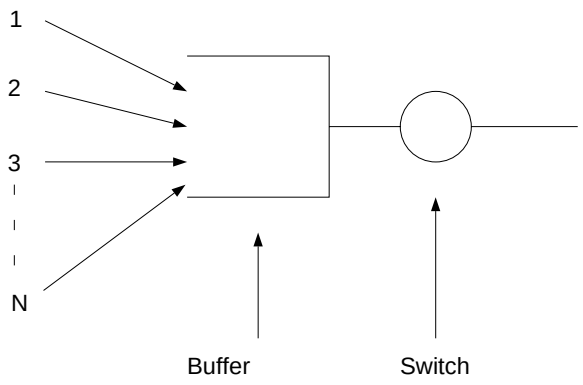


Fig. 9.1. Data streams arriving at a switching element

may result in better reconstructions. Furthermore, instead of selecting motion vectors that match boundaries, those motion vectors that optimize higher order spatial models can be chosen.

In assessing the quality of our reconstructions the mean square error metric was utilized. This metric however, does not accurately predict a human observer's assessment of the reconstructed images. The use of metrics that model the Human Visual System, in evaluating the quality of the reconstructed images are more appropriate. In fact, new error concealment algorithms wherein the objective function to be optimized is based on such metrics would be interesting topics of future research.

In our simulations we utilized 25 streams of MPEG-1 System Layer data. At any network switch, as shown in Figure 9.1, the packets will be carrying data from different applications and of varying bit rates. Congestion at a particular node will thus be a function of N,the number of channels, as well as the bit rates of the various applications. Furthermore, the number of packets dropped per application will depend on how much the network is congested. Thus, the effect of varying N as well as the data rates on the packet loss process and the quality of decoded frames of a particular video sequence, needs to be studied.

In Chapter 8 we described a method whereby unequal error protection and data interleaving was utilized to make the symbols generated by an Embedded Zerotree Wavelet (EZW) encoder resilient to packet loss. It was however observed that the

resulting overhead was large. One way of reducing the overhead is to fix the total number of bits, R_T, used in coding an image. Of the R_T bits, R_S are allocated to the EZW encoder, and R_C set aside for the channel code, i.e. $R_T = R_S + R_C$. The objective is then to minimize the distortion in the decoded image given a particular packet loss rate and that $R_T = R_S + R_C$. Other parameters that constrain the problem are the overall delay between the encoder and the decoder, and decoder buffer fullness. The decoder should receive the coded data within a certain time in order to decode and present it in real-time. This in turn will influence the allowable delays incurred at the encoder and decoder. In addition, the rates R_T, R_S, and R_C should be chosen in such a way to prevent the decoder buffer from underflowing or overflowing. Thus, the ultimate aim is then the judicious allocation and usage of resources, in this case the bit budget and buffers, such that the distortion in the decoded image is minimized. Such a problem can be solved via Lagrangian Relaxation [105], or any other Integer Programming [106] techniques. The problem can be expanded and applied to video coding paradigms such as MPEG where the distortion in the current image being decoded can depend on that of a previously coded image.

LIST OF REFERENCES

[1] M. Tomordy, "Airline with the personal touch," *IEE Review*, vol. 44, no. 6, pp. 261–264, November 1998.

[2] H. Man, F. Kossentini, and M. J. T. Smith, "A class of EZW image coders for noisy channels," *Proceedings of the International Conference on Image Processing*, vol. III, October 26–29 1997, Santa Barbara, California, pp. 90–93.

[3] J. Hagenauer, "Rate compatible punctured convolutional codes (RCPC Codes) and their applications," *IEEE Transactions on Communications*, vol. 36, no. 4, pp. 389–400, April 1988.

[4] Y. Wang and Q. Zhu, "Error control and concealment for video communication: A review," *Proceedings of the IEEE*, vol. 86, no. 5, pp. 974–996, May 1998.

[5] J. Besag, "Spatial interaction and the statistical analysis of lattice systems," *Journal of the Royal Statistical Society,* series B, vol. 36, pp. 192–326, 1974.

[6] R. Kinderman and J. L. Snell, *Markov Random Fields and their Applications*, vol. 1 of *Contemporary Mathematics*. American Mathematical Society, 1980.

[7] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 6, pp. 721–741, November 1984.

[8] M. Wada, "Selective recovery of video packet loss using error concealment," *IEEE Journal on Selected Areas in Communication*, vol. 7, no. 5, pp. 807–814, June 1989.

[9] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Transactions on Signal Processing*, vol. 41, pp. 3445–3462, December 1993.

[10] U. Black, *ATM:Foundation for broadband networks*. Prentice Hall Series in Advanced Communications Technology, Prentice Hall, 1995.

[11] V. Kumar, *Broadband communications*. McGraw-Hill Series on Computer Communications, McGraw-Hill, 1995.

[12] L. G. Cuthbert and J.-C. Sapanel, *ATM:The broadband telecommunications solution.* IEE Telecommunications Series 29, IEE, 1993.

[13] J. Watkinson, *The art of digital video.* Oxford, England: Focal Press, second ed., 1994.

[14] C. P. Sandbank, *Digital Television.* John Wiley and Sons, 1992.

[15] A. M. Tekalp, *Digital Video Processing.* Prentice Hall, 1995.

[16] ISO/IEC 11172 MPEG–1 Standard, *ISO/IEC 11172, Information Technology–coding of moving pictures and associated audio for digital storage media at up to about 1. 5 Mbits/s.* ISO, 1993.

[17] D. H. Pritchard and J. J. Gibson, "Worldwide television standards-similarities and differences," *Journal of the Society for Motion Picture and Television Engineers*, vol. 89, pp. 111–120, February 1980.

[18] K. Jack, *Video Demystified: A Handbook for the Digital Engineer.* Brooktree, 1993.

[19] C. A. Poynton, *A technical introduction to digital video.* John Wiley and Sons, 1996.

[20] *CCIR Recommendation601-3: Encoding Parameters of digital t elevision for studios*, 1992.

[21] S. Prentiss, *High Definition Television.* McGraw Hill, second ed., 1994.

[22] D. LeGall, "MPEG: A video compression standard for multimedia applications," *Communications of the ACM*, vol. 34, no. 4, pp. 47–58, April 1991.

[23] A. K. Jain, *Fundamentals of digital image processing.* Prentice Hall Information and System Sciences Series, Prentice Hall, 1989.

[24] ISO/IEC 11172–2 MPEG–1 Video Coding Standard, *ISO/IEC 11172–2, Information Technology–coding of moving pictures and associated audio for digital storage media at up to about 1. 5 Mbits/s–Part2: Video.* ISO, 1993.

[25] P. Pancha and M. El Zarki, "MPEG coding for variable bit rate video transmission," *IEEE Communications Magazine*, vol. 32, no. 5, pp. 54–66, May 1994.

[26] J. R. Jain and A. K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Transactions on Communications*, vol. COM-29, no. 12, pp. 1799–1808, December 1981.

[27] V. Bhaskaran and K. Konstantinides, *Image and Video Compression Standards: Algorithms and Architectures.* Kluwer Academic Publisher, 1995.

[28] ISO/IEC 13818 MPEG–2 Standard, *ISO/IEC 13818, Generic Coding of Moving Pictures and Associated Audio Information*. ISO, 1995.

[29] J. L. Black, W. B. Pennebaker, C. E. Fogg, and D. J. LeGall, *MPEG Video Compression Standard*. Digital Multimedia Standards Series, Chapman and Hall, 1996.

[30] K. R. Rao and J. J. Hwang, *Techniques and Standards for Image, Video and Audio Coding*. Prentice Hall, 1996.

[31] P.N.Tudor, "MPEG-2 video compression," *Electronics and Communication Engineering Journal*, vol. 7, no. 6, pp. 257–264, December 1995.

[32] ISO/IEC 11172–1 MPEG–1 Systems Standard, *ISO/IEC 11172–1, Information Technology–coding of moving pictures and associated audio for digital storage media at up to about 1. 5 Mbits/s–Part1: Systems*. ISO, 1993.

[33] ISO/IEC 13818–1 MPEG–2 Systems Standard, *ISO/IEC 13818–1, Generic Coding of Moving Pictures and Associated Audio Information–Part1: Systems*. ISO, 1995.

[34] W. Verbiest, L. Pinnoo, and B. Voeten, "The impact of the ATM concept on video coding," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1623–1632, December 1988.

[35] ISO/IEC 13818–2 MPEG–2 Video Coding Standard, *ISO/IEC 13818–2, Generic Coding of Moving Pictures and Associated Audio Information–Part2: Video*. ISO, 1995.

[36] ITU-T, *CCIR Recommendation H. 261: Codec for audiovisual services at px64kbits/sec*, 1990.

[37] ITU-T, *Draft ITU-T Recommendation H.263 Version 2: Video Coding for Low Bitrate Communication*, September 1997.

[38] F. Kishino, K. Manabe, Y. Hayashi, and H. Yasuda, "Variable bit rate coding of video signals for ATM networks," *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 5, pp. 801–806, June 1989.

[39] M. Ghanbari and V. Seferidis, "Cell-loss concealment in ATM video codecs," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 3, no. 3, pp. 238–247, June 1993.

[40] M. Ghanbari and C. J. Hughes, "Packing coded video signals into ATM cells," *IEEE/ACM Transactions on Networking*, vol. 1, no. 5, pp. 505–508, October 1993.

[41] W. Luo and M. El Zarki, "Analysis of error concealment schemes for MPEG-2 video transmission over ATM based networks," *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, vol. 1605, May 1995, Taipei, Taiwan, pp. 1358–1368.

[42] L. H. Kieu and K. N. Ngan, "Cell loss concealment techniques for layered video codec in an ATM network," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 666–677, September 1994.

[43] D. Raychaudhuri, H. Sun, and R. S. Girons, "ATM transport and cell-loss concealment techniques for MPEG video," *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, November 1993, Minneapolis, Minnesota, pp. 117–120.

[44] C. Hahm and J. Kim, "An adaptive error concealment in SNR scalable system," *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, vol. 2501/3, May 24–26 1995, Taipei, Taiwan, pp. 1380–1387.

[45] A. S. Tom, C. L. Yeh, and F. Chu, "Packet video for cell loss protection using deinterleaving and scrambling," *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, May 1991, Toronto, Canada, pp. 2857–2860.

[46] Q. Zhu, Y. Wang, and L. Shaw, "Coding and cell loss recovery in DCT based packet video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 3, no. 3, pp. 248–258, June 1993.

[47] J. Y. Park, M. H. Lee, and K. J. Lee, "A simple concealment for ATM bursty cell loss," *IEEE Transactions on Consumer Electronics*, vol. 39, no. 3, pp. 704–710, August 1993.

[48] Y. Wang, Q. Zhu, and L. Shaw, "Maximally smooth image recovery in transform coding," *IEEE Transactions on Communications*, vol. 41, no. 10, pp. 1544–1551, October 1993.

[49] Y. Wang and Q. Zhu, "Signal loss recovery in DCT-based image and video codecs," *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, vol. 2501/3, November 1991, Boston, Massachusetts, pp. 667–678.

[50] L. T. Chia, D. J. Parish, and J. W. R. Griffiths, "On the treatment of video cell loss in in the transmission of motion-JPEG and JPEG images," *Computers and Graphics: Image Communication*, vol. 18, no. 1, pp. 11–19, January-February 1994.

[51] H. Sun and J. Zdepski, "Adaptive error concealment algorithm for MPEG compressed video," *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, vol. 1818, November 1992, Boston, Massachusetts, pp. 814–824.

[52] W. Kwok and H. Sun, "Multidirectional interpolation for spatial error concealment," *IEEE Transactions on Consumer Electronics*, vol. 3, no. 39, pp. 455–460, August 1993.

[53] H. Sun and W. Kwok, "Concealment of damaged block transform coded images using projections onto convex sets," *IEEE Transactions on Image Processing*, vol. 4, no. 4, pp. 470–477, April 1995.

[54] P. Salama, N. Shroff, E. J. Coyle, and E. J. Delp, "Error concealment techniques for encoded video streams," *Proceedings of the International Conference on Image Processing*, vol. I, October 23–26 1995, Washington, DC, pp. 9–12.

[55] P. Salama, N. Shroff, and E. J. Delp, "A bayesian approach to error concealment in encoded video streams," *Proceedings of the International Conference on Image Processing*, vol. II, September 16–19 1996, Lausanne, Switzerland, pp. 49–52.

[56] P. Salama, N. Shroff, and E. J. Delp, "A fast suboptimal approach to error concealment in encoded video streams," *Proceedings of the International Conference on Image Processing*, vol. II, October 26–29 1997, Santa Barbara, California, pp. 101–104.

[57] P. Salama, N. B. Shroff, and E. J. Delp, "Error concealment in encoded video streams," in *Image Recovery Techniques for Image Compression Applications*, Ed. N. P. Galatsanos and A. K. Katsaggelos, Kluwer Publishers, 1998.

[58] D. C. Youla and H. Webb, "Image restoration by the method of convex projections: Part 1 - theory," *IEEE Transactions on Medical Imaging*, vol. MI-1, no. 2, pp. 81–94, October 1982.

[59] V. Parthasarathy, J. Modestino, and K. S. Vastola, "Design of a transport coding scheme for high quality video over ATM networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 2, pp. 358–376, April 1997.

[60] V. Parthasarathy, J. Modestino, and K. S. Vastola, "Reliable transmission of high quality video over ATM networks," *IEEE Transactions on Image Processing*, vol. 8, no. 3, pp. 361–374, April 1999.

[61] G. Cheung and A. Zakhor, "Joint source/channel coding of scalable video over noisy channels," *Proceedings of the International Conference on Image Processing*, vol. III, September 16–19 1996, Lausanne, Switzerland, pp. 767–770.

[62] Y. Koyama and S. Yoshida, "Error control for still image transmission over a fading channel," *IEEE 45th Vehicular Technology Conference.*, vol. 2, July 25–28 1995, Chicago, IL, pp. 609–163.

[63] P. G. Sherwood and K. Rogers, "Progressive image coding on noisy channels," *Proceedings of the IEEE Data Compression Conference*, March 25–27 1997, Snowbird, Utah, pp. 72–80.

[64] P. G. Sherwood and K. Rogers, "Error protection for progressive image transmission over memoryless and fading channels," *IEEE Transactions on Communications*, vol. 46, no. 12, pp. 1555–1559, December 1998.

[65] P. Salama, N. Shroff, and E. J. Delp, "Error concealment in embedded zerotree wavelet codecs," *Proceedings of the International Workshop on Very Low Bit Rate Video Coding*, October 8–9 1998, Urbana, Il, pp. 200–203.

[66] Requirements, Audio, DMIF, SNHC, Systems, Video, "Overview of the MPEG–4 standard." Stockholm meeting, document ISO/IEC JTC1/SC29/WG11 N1730, July 1997, July 1997.

[67] R. Talluri, "Error resilient video coding in the ISO MPEG-4 standard," *IEEE Communications Magazine*, vol. 36, no. 6, pp. 112–119, June 1998.

[68] J. Liang and R. Talluri, "Tools for robust image and video coding in JPEG2000 and MPEG4 standards," *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, vol. 3653, January 23–29 1999, San Jose, California, pp. 40–51.

[69] S. Wenger, G. Knorr, J. Ott, and F. Kossentini, "Error resilience support in h. 263+," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 7, pp. 867–877, November 1998.

[70] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications.* Prentice Hall, 1983.

[71] S. Wicker, *Error Control Systems for Digital Communication and Storage.* Prentice Hall, 1995.

[72] W. Grimson, "An implementation of a computational theory of visual surface interpolation," *Computer Vision, Graphics, Image Processing*, vol. 22, no. 1, pp. 39–69, April 1983.

[73] R. L. Stevenson, B. E. Schmitz, and E. J. Delp, "Discontinuity preserving regularization of inverse visual problems," *IEEE Transactions on Systems Man and Cybernetics*, vol. 24, no. 3, pp. 455–469, March 1994.

[74] D. Geman and G. Reynolds, "Constrained restoration and the recovery of discontinuities," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 3, pp. 367–382, March 1992.

[75] C. Bouman and K. Sauer, "A generalized gaussian image model for edge-preserving MAP estimation," *IEEE Transactions on Image Processing*, vol. 2, no. 3, pp. 296–310, July 1993.

[76] J. Marroquin, S. Mitter, and T. Poggio, "Probabilistic solution of ill-posed problems in computational vision," *Journal of the American Statistical Association*, vol. 82, no. 397, pp. 76–89, March 1987.

[77] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*. McGraw Hill, third ed., 1984.

[78] R. Schultz and R. L. Stevenson, "A bayesian approach to image expansion for improved definition," *IEEE Transactions on Image Processing*, vol. 3, no. 3, pp. 233–241, May 1994.

[79] P. J. Huber, *Robust Statistics*. John Wiley & Sons, 1981.

[80] J. Besag, "On the statistical analysis of dirty pictures," *Journal of the Royal Statistical Society,* series B, vol. 48, no. 3, pp. 259–302, 1986.

[81] D. G. Luenberger, *Linear and Nonlinear Programming*. Addison Wesley, second ed., 1989.

[82] J. Konrad and E. Dubois, "Bayesian estimation of motion vector fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 9, pp. 910–926, September 1992.

[83] J. Li, X. Lin, and C. C. J. Kuo, "Boundary control vector motion field representation and estimation by using a markov random field model," *Journal of Visual Communication and Image Representation*, vol. 7, no. 3, pp. 230–243, September 1996.

[84] C. E. Bouman, "CLUSTER: An unsupervised algorithm for modeling gaussian mixtures," tech. rep., Purdue University, April 1997.

[85] N. L. A. Dempster and D. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society B*, vol. 39, no. 11, pp. 1–38, 1977.

[86] J. Rissanen, "A universal prior for integers and estimation by minimum description length," *Annals of Statistics*, vol. 11, no. 2, pp. 417–431, 1983.

[87] E. Asbun and E. J. Delp, "Real-time error concealment in compressed digital video streams," *Proceedings of the Picture Coding Symposium*, April 21–23 1999, Portland, Oregon.

[88] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 243–250, June 1996.

[89] C. S. Barreto and G. Mendoncca, "Enhanced zerotree wavelet transform image coding exploiting similarities inside subbands," *Proceedings of the IEEE International Conference on Image Processing*, vol. II, September 16–19 1996, Lausanne, Switzerland, pp. 549–552.

[90] S. A. Martucci, I. Sodagar, T. Chiang, and Y.-Q. Zhang, "A zerotree wavelet video coder," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 7, no. 1, pp. 109–118, February 1997.

[91] K. Shen and E. J. Delp, "Color image compression using an embedded rate scalable approach," *Proceedings of IEEE International Conference on Image Processing*, vol. III, October 26–29 1997, Santa Barbara, California, pp. 69–72.

[92] Q. Wang and M. Ghanbari, "Scalable coding of very high resolution video using the virtual zerotree," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 5, pp. 719–727, October 1997.

[93] K. Shen, *A Study of Real Time and Rate Scalable Image and Video Compression*. PhD thesis, School of Electrical and Computer Engineering, Purdue University, December 1997.

[94] C. D. Creusere, "A new method of robust image compression based on the embedded zerotree wavelet algorithm," *IEEE Transactions on Image Processing*, vol. 6, no. 10, pp. 1436–1442, October 1997.

[95] S. Thillainathan, D. Bull, and N. Canagarajah, "Robust embedded zerotree wavelet coding algorithm," *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, vol. 3309, January 28-30 1998, san Jose, California, pp. 58–99.

[96] J. K. Rogers and P. C. Cosman, "Robust wavelet zerotree image compression with fixed length packetization," *Proceedings of the IEEE Data Compression Conference*, March 30 - April 1 1998, Snowbird, Utah, pp. 418–427.

[97] D. W. Redmill and N. G. Kingsbury, "The EREC: An error-resilient technique for coding variable-length blocks of data," *IEEE Transactions on Image Processing*, vol. 5, no. 4, pp. 565–574, April 1996.

[98] P. J. Sementilli, A. Bilgin, J. H.Kasner, and M. Marcellin, "Wavelet TCQ: submission to JPEG2000," tech. rep., ISO, 1998.

[99] M. W. Marcellin and T. R. Fischer, "Trellis coded quantization of memoryless gauss-markov sources," *IEEE Transactions on Communications*, vol. 38, no. 1, pp. 82–93, January 1990.

[100] V. K. Goyal and J. Kovačević, "Optimal multiple description coding of gaussian vectors," *Proceedings of the IEEE Data Compression Conference*, March 30 - April 01 1998, Snowbird, Utah, pp. 388–397.

[101] R. H. Morelos-Zaragoza and S. Lin, "QPSK block-modulation codes for unequal error protection," *IEEE Transactions on Information Theory*, vol. 41, no. 2, pp. 576–581, March 1995.

[102] R. H. Morelos-Zaragoza and S. Lin, "On primitive BCH codes with unequal error protection capabilities," *IEEE Transactions on Information Theory*, vol. 41, no. 3, pp. 788–790, May 1995.

[103] M. Barazande-Pour, J. W. Mark, and A. K. Khandani, "Multi-level transmission of images over a turbo-coded channel," *Proceedings of the IEEE Pacific-Rim Conference*, August 20–22 1997, Victoria, BC, Canada Utah, pp. 904–907.

[104] H. Ohta and T. Kitami, "A cell loss recovery method using FEC in ATM networks," *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 9, pp. 1471–1482, December 1991.

[105] H. Everett, "Generalized lagrange multiplier method for solving problems of optimal allocation of resources," *Operations Research*, vol. 11, pp. 399–417, 1963.

[106] L. Wolsey, *Integer Programming*. Wiley Interscience, 1998.

VITA

Paul Salama was born in Khartoum, Sudan. In 1991, he received a B.Sc in Electrical Engineering from the University of Khartoum. He was the recipient of the 1988 Shell Company and the University of Khartoum awards for best second year results. He was also awarded the Mighani Hamza award for best final year results in 1991.

In 1993 and 1999 he received an M.S.E.E and a Ph.D in Electrical Engineering, both from Purdue University.

From 1993 to 1995 he was a Research Assistant at the Center for Collaborative Research, and a Teaching Assistant from 1995 to 1999 at Purdue University.

His research interests include image/video compression, image/video restoration, ill posed problems, image reconstruction, and image/video communications over data networks. Paul Salama is a member of Eta Kappa NU, Tau Beta Pi, IEEE, and SPIE.