3-Dimensional Pyramids for Video Compression

Rakeshkumar Hasmukhlal Gandhi, B.Eng.

A THESIS

submitted to the School of Graduate Studies and Research

in Partial Fulfillment of the Requirements

for the Degree of

MASTER OF APPLIED SCIENCE

in

Electrical Engineering

Ottawa-Carleton Institute of Electrical Engineering

Department of Electrical Engineering

Faculty of Engineering

University of Ottawa

OTTAWA, ONTARIO, K1N 6N5

ISBN 0-315-89630-2

Canada

UNIVERSITÉ D'OTTAWA
UNIVERSITY OF OTTAWA

To my family,

Acknowledgments

First, I would like to thank both my supervisors Dr. Morris Goldberg and Dr. Sethuraman Panchanathan for introducing me to the exciting field of video compression and for their support and encouragement during my thesis work. Special thanks are due to Dr. Limin Wang for helpful discussions and for providing the technical materials.

I would also like to thank all the past and current members of the Multimedia Communications Research Laboratory, especially Giridharan Iyengar, Fayez Idris and Dr. M. B. Brahmanandam for their help and co-operation.

My special thanks are also due to all the support staff members of Electrical Engineering for their help, especially Michèle Roy, Suzanne St-Michel, Amanda Lauzon and Lucette Lepage.

I am truly grateful to my beloved wife Varsha and my family for their consistent support, without which this work would not have been possible.

I am thankful to the Canadian Institute for Telecommunications Research (CITR) and the Natural Sciences and Engineering Research Council (NSERC) of Canada for their financial support and the Communications Research Center for the computing facilities.

Abstract

The larger memory and channel bandwidth requirements for digital video transmission and storage make it mandatory to use compression techniques. Representation of the video signal in a pyramid format not only compresses the signal but also makes it suitable for specific applications such as packet video based on asynchronous transmission mode (ATM) and compatible advanced television (ATV). In addition, an efficient bit rate control can be achieved by coarsely quantizing the final level of the pyramid.

In this thesis, we propose to employ a pyramid data structure for video compression. A review of video coding schemes is first presented, followed by a review of the various 2-dimensional (2D) and 3-dimensional (3D) pyramid data structures from the perspectives of data compression. The performance of different configurations of temporal/spatial pyramid data structures is then measured for video compression in terms of the first order entropy. Based on this study, we introduce an efficient 3D adaptive temporal/spatial pyramid which selects either the temporal or spatial contractions using the temporal and spatial prediction differences, respectively. We propose a video codec that combines the adaptive temporal/spatial pyramid and an intra-frame coding technique. The bits are allocated to different levels of the pyramid using a simple buffer control scheme. Lossless coding is achieved by feeding-forward the encoding errors introduced at the upper levels to the lower levels of the pyramid. The proposed pyramidal video codec has a number of similarities with the MPEG video compression standard namely use of bidirectional motion estimation and concepts of I, P and B frames.

Simulation results on CCITT standard video sequences indicate that the adaptive pyramid reduces the lossless bit rate by a factor of two. For video conferencing applications, excellent subjective quality as well as objective quality (PSNR value of 36.6 db) are obtained at a bit rate less than T1 rate (i.e. 1.544 Mbits/s). Promising results have been obtained for CCIR resolution (720 × 480), high detail sequences at a bit rate of 6 Mbits/s. Furthermore, smooth transition is achieved in the case of scene changes without sacrificing picture quality. Finally, the algorithm is well suited for constant bit rate and constant quality applications.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Video Compression

There is an increasing demand for information in visual form in many applications including Multimedia Communications, High Definition Television (HDTV), Telepresence, etc. There is a growing move from the analog to digital TV. A digital signal has many advantages over an analog signal in terms of the processing flexibility, robustness to channel errors and high signal to noise ratio (SNR). For a digital signal, the large channel bandwidth required for transmission and memory required for storage necessitates the use of compression techniques, irrespective of advent of broadband networks and advances in storage technology. For example, a color video sequence in Common Intermediate Format (CIF, 288 $\times$ 352) for video conferencing application has a total data rate of 46 Mbits/s. The real-time transmission of such video sequence over the T1 rate (1.544 Mbits/s) channel implies a compression ratio of 30:1.

The natural video signal has a very high degree of correlation in both the temporal (inter-frame) and spatial (intra-frame) domains. Most of the existing techniques attempt to exploit these correlations and remove the redundancies using combinations of inter/intra-frame coding techniques [3]. In addition, limitations of the human visual system (HVS) which cannot perceive high spatial frequencies can be exploited to achieve perceptually transparent image coding [81].

Recently, the Joint Photographic Experts Group (JPEG) [15] and the Motion Pictures Experts Group (MPEG) [20] of the International Standards Organization (ISO) have proposed standards for still image and video compression, respectively. In addition, the International Consultative Committee on Telephony and Telegraphy (CCITT) [18, 19] has recommended an algorithm for video compression, known as the H.261 standard, at bit rates of $p \times 64$ Kbits/s, where p is in the range 1 to 30. Hardware implementations of these standard video codecs are widely available [25, 24].

Various applications require video sequences at different raster sizes and frame rates (refresh rates). There is a great demand for a generic coder that can provide an output video signal at different resolutions. For example, a browsing application requires the information in progressive form where the gross information is transmitted first. Multiresolution data structures address this problem by reorganizing the data into different resolutions [59].

1.2 Investigated Approach

In this thesis, the problem of video compression is addressed from two perspectives, reorganization of the signal into a pyramid data structure and encoding the pyramid data structure. Algorithms to form the temporal/spatial pyramid data structures are presented. The encoding techniques for the pyramids, namely, error-delivery, adaptivity control and bit allocation method are then described.

The first step in compressing a video signal is to represent the signal in a proper format before it is coded. Proper representation of the signal not only compresses the data, but also makes it suitable for specific applications. For example, a pyramidal representation [59, 62] of the video signal is useful in applications such as packet video based on asynchronous transmission mode (ATM) (by dividing significant and non-significant information in the pyramid) and compatible advanced television (ATV). In addition, an efficient bit rate control can be achieved by coarsely quantizing the final level of the pyramid.

The temporal and spatial correlations present in the video signal can be exploited adaptively, i.e. the temporal or spatial coding techniques can be selected based on the correlations in the corresponding domains. For example, in case of fast motion in the video sequence, higher compression ratios can be obtained if spatial coding techniques are employed [49]. In this thesis, an adaptive temporal/spatial pyramid data structure

is proposed which effectively exploits the temporal and spatial correlations. This pyramid is used in a "reduced" form and does not require transmission of any overhead information. In addition, bidirectional motion compensated temporal prediction and non-causal spatial prediction can be incorporated into the adaptive pyramid.

A video codec is designed using the proposed adaptive pyramid and an intra-frame coding technique. A control signal is generated using the displaced frame difference obtained from motion estimation to select either the temporal or spatial contraction. A hierarchical buffer control scheme is detailed in combination with a bit allocation method. Algorithms are presented for error-delivery in temporal and spatial domains. Note that the error-delivery technique provides the posssiblity of lossless coding even though a lossy coder is employed at the upper levels of the pyramid. The proposed pyramidal video codec has a number of similarities with the MPEG video compression standard namely use of bidirectional motion estimation and concepts of I, P and B frames.

1.3 Thesis Organization

The thesis is organized as follows. In chapter 2, a review of video compression techniques is presented. First, we detail entropy, rate distortion function and performance measures. This is followed by a brief review of lossless and lossy image coding techniques. We review two lossless coding techniques, Huffman coding and run-length coding. Three basic lossy techniques; predictive coding, transform coding and vector quantization (VQ) are described. We then present video compression schemes which use combinations of inter/intra-frame coding techniques. Finally, buffer control strategies which are important in a low bit rate video coding are reviewed.

In chapter 3, algorithms to form 2-dimensional (2D) and 3-dimensional (3D) pyramid data structures are described from the perspectives of data compression. We

review the mean pyramid, Gaussian pyramid, S-transform pyramid and sum pyramid. The difference pyramids are used for coding purposes as the correlation in the pyramid is reduced. The algorithms to construct the difference pyramid, reduced difference pyramid and the Laplacian pyramid are presented. Two methods of forming 3D pyramids called the *spatio-temporal pyramid* and the *3D Laplacian pyramid* are explained.

In chapter 4, algorithms to form the temporal and spatial sample pyramids are presented. We discuss methods to form the difference pyramids using temporal bidirectional motion compensated prediction and spatial non-causal linear prediction. Performance comparisons of the various temporal/spatial pyramid data structures for video compression in terms of the first order entropy are then presented. Based on this study, we propose an efficient 3D adaptive temporal/spatial prediction difference pyramid. We show that the adaptive pyramid achieves the lowest first order entropy.

In chapter 5, the design of a video codec using the 3D adaptive pyramid and an intra-frame coding technique is described. We present algorithms for the temporal and spatial error delivery and show that the coder can be used for lossless coding. Then a hierarchical buffer control scheme which uses a bit allocation method is detailed. The bit allocation method assigns bits to the different levels of the pyramid such that the quality of the frames are constant while we retain a simple buffer control mechanism. The simulation results are reported for monochrome CCITT standard test sequences. We compare the performance of the proposed pyramidal coder with that of the MPEG coder. Finally, the main features of the video coder are outlined.

1.4 Main Contributions

The main contributions of this thesis are summarized below:

- Present performance comparison of temporal/spatial pyramids for video compression.

- Introduce a novel 3D adaptive pyramid data structure that achieves the lowest first order entropy.

- Propose a video codec which combines the adaptive pyramid and an intra-frame coding technique.

- Describe a hierarchical buffer control scheme in combination with a bit allocation method.

Chapter 2

Review of Video Compression Techniques

In this chapter, first, a review of entropy, rate distortion function and performance measures is presented. This is followed by a brief review of the lossless and lossy image coding techniques, which include Huffman coding, run-length coding, predictive coding, transform coding, vector quantization. Finally, hybrid video compression schemes which use combinations of the different inter/intra-frame coding techniques are described.

2.1 Information Theory

In information theory, the concepts of entropy and rate distortion are important. Entropy is useful measure of information because it gives the minimum average bit rate for perfect reconstruction. Rate distorion theory provides a lower bound on the average bit rate for a given distortion and hence an upper bound on the performance of waveform coders. The concepts of entropy and rate distortion are detailed in the following sections.

2.1.1 Entropy

An image source, X, of size $N \times N$, with each pixel quantized to K gray levels can generate a total of $K^{N \times N}$ possible image patterns. Let the probability of a specific image pattern be given by $p(x)$ where

$$x = x_{ij}, \quad i,j = 0,1,\cdots,N-1, \tag{2.1}$$

where x_{ij} is the $(i,j)th$ element of x. The average information of the source is specified by its entropy which is defined as [11],

$$H(X) = -\frac{1}{N \times N} \sum_{all\ x} p(x) \log_2 p(x) \quad bits/pixel. \tag{2.2}$$

It has been shown that [1] the source entropy is lower bounded by 0 and upper bounded by $\log_2 K$, that is,

$$0 \leq H(X) \leq \log_2 K. \tag{2.3}$$

The left side equality holds if all probabilities except one are zero, in which case the source is totally predictable. The right side equality holds when every source symbol has an equal probability. The redundancy of the source is given as,

$$redundancy = \log_2 K - H(X). \tag{2.4}$$

If every pixel in an image were statistically independent of the others, then the source probability $p(x)$ can be expressed as,

$$p(x) = \prod_{i,\ j=0}^{N-1} p_{ij}(x_{ij}). \tag{2.5}$$

Here, p_{ij} represents the probability that the pixel X_{ij}, of the image source X, has a value equal to x_{ij}. In this case we get $H(X)$ as,

$$H(X) = -\frac{1}{N \times N} \sum_{i,j=0}^{N-1} \sum_{all\ x_{ij}} p_{ij}(x_{ij}) \log_2 p_{ij}(x_{ij}). \tag{2.6}$$

In practice, the statistical information of an image, $p(x)$, is, however, not easily measured or modelled and therefore, the true entropy of the image is, in general, very difficult to obtain. Hence, a simpler measure, the first order entropy $H^1(X)$, which is defined on a pixel-by-pixel basis is often used. $H^1(X)$ can be defined as,

$$H^1(X) = -\sum_{k=0}^{N-1} P_k \log_2 P_k \tag{2.7}$$

where P_k is the probability of occurence of gray level k. If the pixels of the image are identically and independently distributed (i.i.d), that is,

$$p_{ij}(x_{ij}) = p_{uv}(x_{uv}), \quad for\ x_{ij} = x_{uv} \tag{2.8}$$

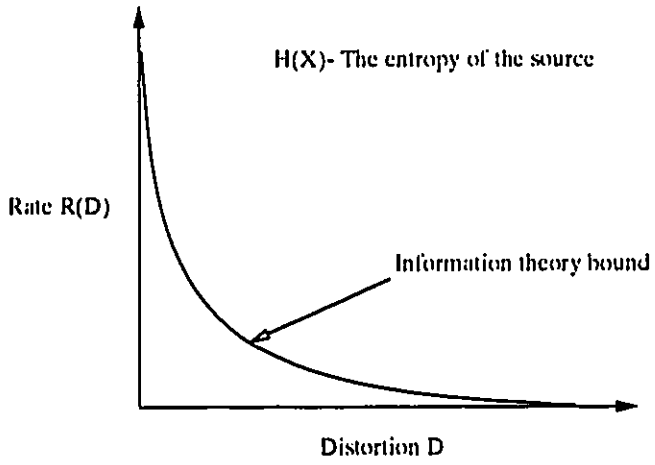

Figure 2.1: An example of the rate distortion function for a discrete-amplitude source

then the entropy $H(X)$ of the image equals the first order entropy, $H^1(X)$,

$$H(X) = H^1(X) = -\sum_{k=0}^{N-1} P_k \log_2 P_k. \tag{2.9}$$

In this case, P_k is simply the probability of the occurence of the kth gray level value. The first order entropy is often called the *memoryless entropy.* The higher order entropies are defined in [10]. In this thesis, the first order entropy is used to measure the compactness of the pyramid data structures for video compression. We note that first order entropy gives the minimum bit rate for lossless reproduction of the video signal.

2.1.2 Rate Distortion Function

For lossy coding, the rate distortion function (RDF) determines the minimum distortion that can be achieved for a given average bit rate [11]. Hence the RDF provides an upper bound on the performance of practical coders as shown in Fig. 2.1. The RDF is a measure of transmission of information from the source to the receiver.

The average mutual information, $I_{N\times N}(X,\hat{X})$, between the source and the receiver is defined as,

$$I_{N\times N} = \frac{1}{N\times N}\sum_{all\ x,\hat{x}} p(x)p(\hat{x}/x)\log_2 \frac{p(\hat{x}/x)}{\sum_{all\ x} p(x)p(\hat{x}/x)}. \tag{2.10}$$

This is a measure of the statistical dependence between the source output X and the decoded output $\hat{X}$. Since $p(x)p(\hat{x}/x) = p(x,\hat{x})$, it follows that

$$\sum_{all\ x} p(x)p(\hat{x}/x) = \sum_{all\ x} p(x,\hat{x}) = p(\hat{x}) \tag{2.11}$$

and therefore,

$$I_{N\times N} = \frac{1}{N\times N}\sum_{all\ x,\hat{x}} p(x,\hat{x})\log_2 \frac{p(\hat{x}/x)}{p(\hat{x})}. \tag{2.12}$$

We can show that

$$I_{N\times N} = H(X) - H(X/\hat{X}) \tag{2.13}$$

where $H(X/\hat{X})$ is the entropy of the source X, given the decoder output $\hat{X}$. Hence the mutual information between X and $\hat{X}$ is the difference in the uncertainty at the source X and the uncertainty at the source X, given the decoder output $\hat{X}$. If the encoding is lossless then the average mutual information is given by,

$$I_{N\times N} = H(X) - H(X/\hat{X}) = H(X). \tag{2.14}$$

This implies that for lossless coding, the least average bit rate is $H(X)$. On the other hand, if there is no information between the encoder and the decoder, the mutual information $I_{N\times N}$ is equal to 0. Therefore, the average mutual information is lower bounded by 0 and upper bounded by the source entropy, i.e.

$$0 \le I_{N\times N} \le H(X). \tag{2.15}$$

If we define a distance or distortion measure $d(x,\hat{x})$ between x and $\hat{x}$, the average distortion per pixel is therefore,

$$D = \frac{1}{N\times N}E[d(X,\hat{X})] = \frac{1}{N\times N}\sum_{all\ x,\hat{x}} d(x,\hat{x})p(x)p(\hat{x}/x). \tag{2.16}$$

The $N \times N$ block rate distortion function $R_{N\times N}(D)$ [11] is defined as the minimum average mutual information between X and $\hat{X}$ subject to the constraint of a fixed average distortion D,

$$R_{N\times N}(D) = \inf_{all\ p(\hat{x}/x)} I_{N\times N}(X, \hat{X}). \tag{2.17}$$

Note that since the source is known, the minimization can only be over the conditional probability $p(\hat{x}/x)$. The rate distortion function $R(D)$ [11] is simply the limiting value of the $N \times N$ - block rate distortion function $R_{N\times N}(D)$ as the block size $N \times N \rightarrow \infty$,

$$R(D) = \lim_{N\times N\rightarrow\infty} R_{N\times N}(D). \tag{2.18}$$

In other words, the distortion D and the mutual information $I_{N\times N}(X, \hat{X})$ depend on the type of source coding. However, there is a minimum $I_{N\times N}$ that is needed so that the average distortion of reconstruction at the destination does not exceed the specified upper limit D. This minimum value of $I_{N\times N}$ is $R(D)$. For perfect reconstruction (D = 0), when complex source coder is employed, the bit rate required is just equal to the source entropy or average self-information, i.e.

$$R(0) = H(X). \tag{2.19}$$

2.1.3 Distortion Measures

A distortion measure is used to evaluate the performance of any source coding scheme, and it essentially is a cost function $d(x, \hat{x})$, for reproducing the input x by an output $\hat{x}$. With such a cost function, the performance of a coding system can be evaluated using the average distortion introduced by the coding system, $E[d(X, \hat{X})]$,

$$E[d(X, \hat{X})] = \sum_{all\ x,\hat{x}} d(x, \hat{x}) p(x, \hat{x}). \tag{2.20}$$

To be of value for both design and comparison, a distortion measure must be,

1. Computable, so that it can be efficiently evaluated in real time;

2. Subjectively meaningful, so that it correlates well with human visual observations;

3. Tractable, so that it is mathematically analyzable.

One of the most widely used distortion measures in image coding is, the Peak Signal to Noise Ratio (PSNR) [2], defined as,

$$PSNR = 10\log_{10}(\frac{255 \times 255}{MSE}) \tag{2.21}$$

where MSE is the Mean Square Error, defined as,

$$MSE = \frac{1}{M \times N}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(x_{ij} - \hat{x}_{ij})^2 \tag{2.22}$$

The other distortion measures are, the Normalized Mean Square Error (NMSE), Signal to Noise Ratio (SNR) and Mean Absolute Error (MAE) defined as follows:

$$NMSE = \frac{\left[\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(x_{ij} - \hat{x}_{ij})^2\right]}{\left[\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(x_{ij})^2\right]} \tag{2.23}$$

$$SNR = 10\log_{10}(\frac{1}{NMSE}) \tag{2.24}$$

$$MAE = \frac{1}{M \times N}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(|x_{ij} - \hat{x}_{ij}|) \tag{2.25}$$

where $(|x_{ij} - \hat{x}_{ij}|)$ is the absolute difference of the x_{ij} and $\hat{x}_{ij}$. Note that x_{ij} is the original image pixel and $\hat{x}_{ij}$ is the reconstructed image pixel.

Each of the objective error criteria has advantages and disadvantages. Neither the MSE nor the MAE correlate very well with the subjective ratings. For example, two different images with the same PSNR values may receive very different subjective evaluations. There is a considerable effort in the research community to define newer measures which yield higher correlation with the subjective observations [1].

In this thesis, the PSNR is used as the standard criterion for evaluating different schemes for video compression. This measure is analytically tractable and computable in real-time. The low and high PSNR values correspond to low and high subjective quality, respectively.

2.2 Lossless Coding

In natural images, there exists a high degree of correlation between the neighbouring pixels, implying statistical redundancy in the image. In lossless coding, this redundancy is exploited in such a way that the process in reversible; i.e. the original image is recovered exactly. There is a lot of interest in lossless techniques, especially in applications such as medical imaging [14]. In this section, two lossless techniques namely *Huffman coding,* and *Run-length coding* are briefly reviewed. In this thesis, an entropy coding scheme using a combination of run-length and Huffman coding is employed as in the MPEG coder.

2.2.1 Huffman Coding

From Shannon's first theorem (lossless coding theorem), we know that the average bit rate R for encoding a source cannot be less than the source entropy H. Huffman coding [12] provides a practical procedure for lossless encoding which tries to achieve the Shannon limit. The Huffman procedure results in a variable length code (VLC) in which the number of bits for a source sample varies approximately as the inverse of the probability of that sample. This coding technique is instantly decodable. If the efficiency of the code is defined as,

$$Efficiency = \frac{H}{R} \times 100\% \tag{2.26}$$

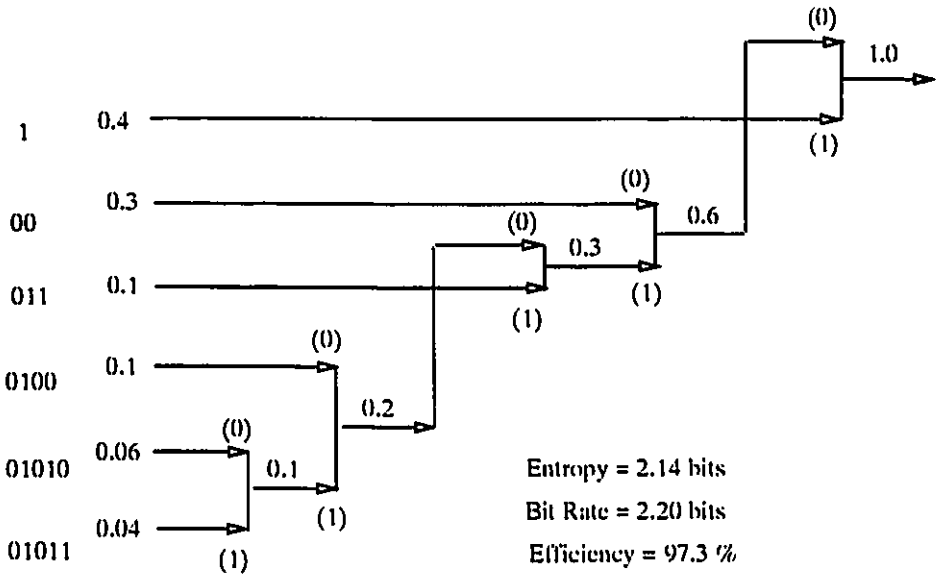

Figure 2.2: An example showing the construction of a Huffman coding tree

then Huffman code achieves the Shannon bound and an efficiency of 100% when the source probabilities are negative powers of 2.

The method to build a Huffman code table for a given source is illustrated in Fig. 2.2. First, the probabilities of the source are arranged in decreasing order (0.4, 0.3, 0.1, 0.1, 0.06, 0.04). The two smallest probabilities (0.06, 0.04) are then combined to yield a new single value, resulting in a reduced set of probabilities. This is again re-arranged in decreasing order (0.4, 0.3, 0.1, 0.1, 0.1). The process is repeated until a set containing only one value (unity) is reached. Each node of the binary tree corresponds to the new probability formed, a binary number '0' is assigned to the upper member and '1' to the lower as shown in Fig. 2.2. The codeword for a source sample is the sequence of 0's and 1's obtained by tracing the path from the leaf of the tree to the root of the tree.

The design of a Huffman code tree for a given source depends upon the source statistics. If the source statistics change, a new code tree has to be built for efficient

coding. To tackle this problem, an adaptive Huffman is used to build the code tree in which the encoder tracks the probability of each source symbol as it occurs [2]. If both the transmitter and the receiver start with the same Huffman codes and modify them using the same algorithm, then perfect reconstruction is guaranteed without transmission of any side information.

2.2.2 Run-length Coding

Another popular lossless coding technique is the run-length coding (RLC) scheme. This technique is extensively used in facsimile transmissions. A run is defined as a sequence of consecutive pixels of identical values along a specified direction, for example, the horizontal scan line. If the runs are long, significant bit rate reduction can be achieved. Run-length coding can be very efficient for binary images such as graphics, text, handwritten material, etc. For detailed images such as natural images, the scheme is not as efficient. If natural images are split into a set of bit planes and then run-length coding is applied, significant compression can be achieved. In transform coders, the coefficients below some threshold value are set to zero. The resulting transform matrix contains long run of zeros. Hence, run-length coding is efficiently used in transform coding [15].

Run-length coding can be extended to two dimensions. In area coding, an area is defined as a connected, continuous group of pixels of identical value. To transmit an image, only the values which specify the area and the intensity of the area are transmitted.

2.3 Lossy Coding

The lossless coding schemes are not suitable in applications where high compression ratios are required. In practice, lossy coding techniques are employed in such applications. In lossy coding, the objective is to reduce the transmission bit rate subject to some constraints on image quality. Lossy coding techniques fall into one of the two basic categories: predictive coding and transform coding [2, 1]. Predictive coding methods exploit *redundancy* related to the predictability, randomness and smoothness in the data. For example, an image of constant gray level is fully predictable once the gray level of the first pixel is known. In transform coding, data is transformed from spatial to frequency domain where a large amount of information is packed into a small number of samples, called the *coefficients*. Both methods have relative advantages and limitations. For example, predictive coding systems have a much lower complexity and memory requirement. However, at low bit rates, the compression ratios obtained are not high. In transform coding, higher compression ratios can be achieved at the expense of computational complexity and memory usage. We review predictive and transform coding, since both techniques will be used in the thesis. Another promising technique for low bit rate image compression is vector quantization (VQ). Shannon's rate distortion theory states that better performance can be achieved by coding vectors instead of scalars [11]. In VQ, an input vector is mapped into an output vector using the predesigned codebook of vectors. Thus, higher compression ratios are possible using the VQ techniques. Details of the VQ are presented in section 2.3.3.

2.3.1 Predictive Coding

In predictive coding, the data to be encoded is first predicted from previously available data using prediction rules. The predicted estimate is then subtracted from the actual

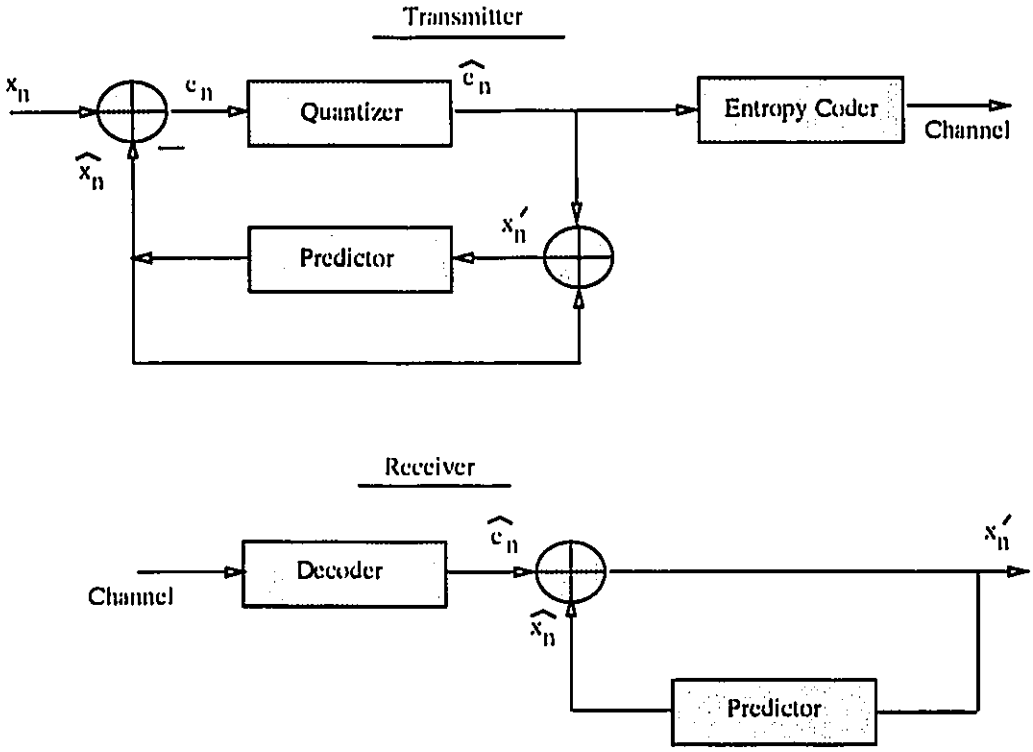

Figure 2.3: Block diagram of the DPCM system

data and the difference is then quantized, entropy coded, and transmitted. This method is called the *differential pulse code modulation (DPCM).*

DPCM

The block diagram of the DPCM system is shown in Fig. 2.3. The difference between the original pixel, x_n, and its predicted pixel, $\hat{x}_n$, called e_n is quantized and entropy coded. The quantized information is in turn used to predict the next pixel. The prediction rule can be linear or non-linear. In linear predictive coding [1], the predicted value $\hat{x}_n$ is calculated for each input pixel, x_n, by using the following equation,

$$\hat{x}_n = \sum_i a_i x_{n-i} \tag{2.27}$$

The predictor can be optimized in terms of the MSE. In other words, the coefficients a_i can be selected such that the variance or the energy of the prediction error is minimized. The optimal set of coefficients can be obtained from the following equation,

$$\sum_i a_{i,opt} R(j-i) = R(j) \tag{2.28}$$

where $R(i)$ is the correlation function. The prediction difference can be quantized depending on the available data rate and MSE requirements. *Delta Modulation* is a special case of DPCM, where the quantizer has only two levels, and if the difference is positive, it is $+\Delta$ coded, otherwise, $-\Delta$ coded. Here, Δ is the predetermined output value.

Adaptive DPCM

DPCM can exploit the local correlation efficiently. However, it is quite sensitive to changes in the statistics of the data. In case of non-stationary data such as images, a fixed predictive coder may not yield consistent performance in terms of the rate distortion function. Adaptive techniques can be employed in order to match the variations in the image statistics. The adaptive approaches fall into one of two classes: an adaptive predictor with a fixed quantizer or an adaptive quantizer with a fixed predictor [6]. In these adaptive approaches, the parameters involved in designing the predictor and the quantizer are required to keep up with the changing statistics of the input data. For example, in the case of a linear predictor, the correlations and the corresponding weighting coefficients are recomputed periodically [6].

2.3.2 Transform Coding

The block diagram of a transform coding scheme is shown in Fig. 2.4. Here, the

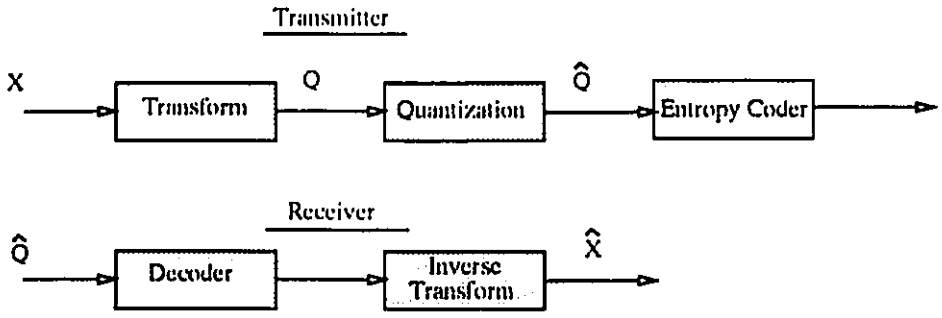

Figure 2.4: Block diagram of the transform coding system

input image, X, of size $N \times N$ first undergoes an orthogonal transform T,

$$Q = TXT' \tag{2.29}$$

where, $T' = T^{-1}$. The resulting decorrelated coefficients Q_{ij}, $i, j, \cdots, N-1$ of the transformed image are then quantized with a variable number of bits, the number being specified by a *bit allocation map*, giving $\hat{Q}$. At the receiver, the inverse transform is performed to recover the coded image, $\hat{X}$, that is,

$$\hat{X} = T'\hat{Q}T. \tag{2.30}$$

Transformation

A transform which minimizes the geometric mean of the variances of the transform coefficients is called an *optimal transform* [1]. Hence, an optimal transform completely decorrelates the transform coefficients. Minimizing the geometric mean of the variances of the transform coefficients corresponds to minimizing the reconstruction error variance in the transform domain. One example of an optimal orthogonal transform is the discrete Karhunen-Loeve transform (KLT). The implementation of KLT requires a large amount of computations. Hence, in practice, suboptimal transforms such as sine, cosine, Fourier, Hadamard, etc. are employed. It has been demonstrated

that Discrete Cosine Transform (DCT) performs closest to the KLT, especially when there is a high correlation in the data. DCT has been chosen as an intra-frame coding scheme in the MPEG, JPEG and H.261 standard algorithms.

Orthogonal transforms have following two interesting properties:

1. The average energy of the transformed image is equal to the average energy of the input image, that is,

$$\frac{1}{N \times N} \sum_{i,j=0}^{N-1} \sigma_{ij}^2 = \frac{1}{N \times N} \sum_{i,j=0}^{N-1} \sigma_{x,ij}^2 = \sigma_x^2. \tag{2.31}$$

   Here $\sigma_{ij}^2 = E(Q_{ij}^2)$, $i,j = 0,1,\cdots,N-1$ is the second moment of the (i,j)th input element.

2. The average reconstruction error variance is equal to the average quantization error variance in the transform domain, that is,

$$\sigma_r^2 = \frac{1}{N \times N} \sum_{i,j=0}^{N-1} \sigma_{r,ij}^2 = \frac{1}{N \times N} \sum_{i,j=0}^{N-1} \sigma_{q,ij}^2 = \sigma_q^2. \tag{2.32}$$

   Here $\sigma_{r,ij}^2 = E((X_{ij} - \hat{X}_{ij})^2)$, $i,j = 0,1,\cdots,N-1$ is the reconstruction error variance of the (i,j)th input element and $\sigma_{q,ij}^2 = E((Q_{ij} - \hat{Q}_{ij})^2)$, $i,j = 0,1,\cdots,N-1$, is the quantization error variance of the (i,j)th transform coefficient.

Eqns. 2.31 and 2.32 imply that the analysis in the spatial domain will be completely reflected in the transform domain in terms of the average value of the second moment.

Quantization

In terms of scalar quantization, the quantization error variance σ_q^2 can be related to the input signal variance σ_x^2 as follows,

$$\sigma_q^2 = \epsilon_q^2 \sigma_x^2 = \epsilon^2 2^{-2B} \sigma_x^2. \tag{2.33}$$

Here, $\epsilon_q = \epsilon^2 2^{-2B}$ is the quantizer performance factor which depends on the probability density function (PDF) of the input signal and on the quantizer characteristics, such as number of quantization levels, etc. The quantity ϵ^2 can be considered as a variable correction factor that takes into account the performance of a practical quantizer and B is the number of bits assigned to the quantizer.

Bit Allocation

There is little gain by using transform coding in terms of compression if equal bits are assigned to all the coefficients in the transform domain [1]. Hence, it is required that the bits be allocated to the coefficients based on some method such that significant coefficients get higher bits and vice versa. An optimum bit allocation is given by,

$$\begin{aligned} Bij &= B + \tfrac{1}{2}\log_2 \epsilon_{ij}^2\sigma_{ij}^2 - \tfrac{1}{N\times N}\textstyle\sum_{k,l=0}^{N-1} \tfrac{1}{2}\log_2 \epsilon_{kl}^2\sigma_{kl}^2 \\ &= B + \tfrac{1}{2}\log_2 \frac{\epsilon_{ij}^2\sigma_{ij}^2}{[\prod_{k,l=0}^{N-1}\epsilon_{kl}^2\sigma_{kl}^2]^{1/(N\times N)}} \qquad i,j = 0,1,\cdots,N-1 \end{aligned} \tag{2.34}$$

It should be observed that the number of quantizer levels $2^{B_{ij}}$ is proportional to both the coefficient variance, σ_{ij}^2, and the quantizer performance factor, ϵ_{ij}^2.

Adaptive Transform Coding

Chen *et al* [44] have proposed an adaptive transform coding scheme in which the bit allocation map changes to match the image statistics. In this technique, each block is classified into one of several classes based on its activity. A larger number of bits is then assigned to the blocks with higher activity and fewer bits to the blocks with lower activity. We note that this technique is often used for comparing coding performances and has become a part of the JPEG, MPEG and H.261 standards.

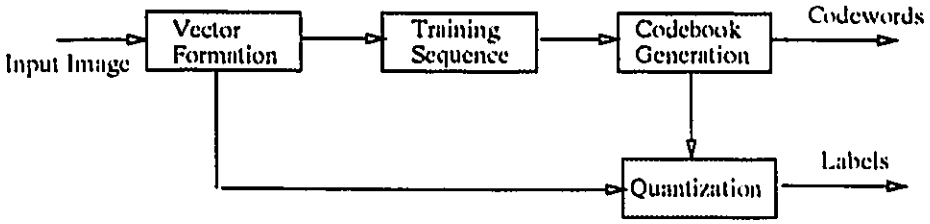

Figure 2.5: Block diagram of the vector quantizer

2.3.3 Vector Quantization

Vector quantization (VQ) is an efficient technique for very low bit rate image coding [63, 65]. VQ can be defined as follows: An N-level, L-dimensional vector quantizer is a mapping, q, that assigns to each input vector, $X = \{X_0, \ldots, X_{L-1}\}$, a reproduction vector, $Y = q(X)$, drawn from a finite reproduction alphabet, $C = \{\Re_i, i = 1, \ldots, N\}$,

$$q(X) = R^n \rightarrow C \tag{2.35}$$

The quantizer q is completely described by the reproduction alphabet (also called *codebook*) C together with partition, $P = \{P_i, i = 1, \ldots, N\}$ where $P_i = \{X : q(X) = Y_i\}$ is an input vector mapping into the ith reproduction vector called *codeword*.

If $d(X_i, Y_j)$ is a distortion measure which represents the error when X_i is reproduced by Y_j, then the optimal quantizer is the one which minimizes $d(X_i, Y_j)$ over all other quantizers [64]. The quantizer performance can be quantified by the average distortion,

$$\begin{aligned} D &= E\{d(X,Y)\} \\ &= \sum_{i=1}^{N} p(X \epsilon \Re_i) E\{d(X,Y_i))|X \epsilon \Re_i\}. \end{aligned} \tag{2.36}$$

It follows that the optimal quantizer must satisfy the following conditions:

1. The selection rule should be a minimum distortion or nearest neighbour rule;

i.e.

$$q(X_i) = Y_j \quad iff \quad d(X_i, Y_j) < d(X_i, Y_k) \quad for\ all\ k \tag{2.37}$$

where $q(.)$is the quantization operation.

2. The codewords $\{Y_i, i = 1, 2, \cdots, N\}$ are determined such that the average distortion in each region is minimized (i.e. the codewords are the centroids of the decision).

VQ consists of two steps: quantization and decoding [63]. In quantization, the first step is to decompose the input image into a set of vectors. For each input vector of dimension L, the predesigned codebook is searched to obtain the closest codeword. Compression is achieved by transmitting the index (label) of the codeword. Decoding of the images can be implemented by simple table look-up techniques where the label is used as an address to a table containing the codewords. The codebook can be broadly classified into universal or image adaptive, resulting in universal VQ (UVQ) or image adaptive VQ (IAVQ) [65]. In the UVQ, the same copy of the codebook is kept at the transmitter and the receiver and hence, does not require the transmission of the codebook. In IAVQ, the codebook is generated on the fly from the image to be coded and transmitted to the receiver as side information [72, 71]. Although IAVQ has some overhead in terms of bit rate and computational complexity, adaptive codebooks match the image statistics better resulting in a superior coding performance. In this thesis, the UVQ technique is used where the codebooks are generated using the LBG algorithm described in the following section.

LBG Algorithm for Codebook Design

Linde *et al* [64] have proposed an algorithm for codebook design based on the two conditions for optimality, referred to as the generalized Lloyd or the LBG algorithm.

In this algorithm, given an initial codebook, each training vector is assigned to its nearest neighbour codeword. Each codeword is then modified to minimize its distortion relative to the vectors assigned to it. This process continues iteratively until the change in distortion between two successive iterations is sufficiently small. The algorithm is described as follows.

1. Given an initial codebook, $C_0 = \{Y_i, i = 1, 2, \cdots, N\}$, a threshold $\delta \geq 0$, and a training set $\{X_i, i = 1, 2, \cdots, K\}$, set m to 0 and D_{-1} to ∞.

2. Assign each input vector to its nearest neighbour codeword,

$$X_i \in Y_{j,m} \quad iff \quad d(X_i, Y_{j,m}) \leq d(X_i, Y_{k,m}) \quad for\ all\ j \neq k \tag{2.38}$$

3. Find C_{m+1}, by computing the centroids of the training vectors assigned to each codeword,

$$Y_{i,m+1} = \frac{1}{M_{i,m}} \sum_{X_j \in Y_{j,m}} X_j \qquad i = 1, 2, \cdots, N \tag{2.39}$$

   where $M_{i,m}$ is the number of vectors assigned to $Y_{j,m}$.

4. Compute the average distortion,

$$D_m = \frac{1}{N} \sum_{i=1, Y \in C_m}^{N} D(X_j, Y) \tag{2.40}$$

   if D_m relative to D_{m-1} is less than δ, then stop; otherwise, go to step 2.

To obtain an initial codebook C_0, one possible approach is to select the evenly spaced N vectors as an initial codebook. Alternatively, one might use the splitting algorithm [64], where the centroids of the training set are clustered and split into two codewords. The LBG algorithm is applied to yield a codebook of two codewords. Each codeword is then split into two codevectors to yield a codebook of four codewords. This procedure is repeated until N level codebook is generated.

Universal Vector Quantization

Universal vector quantization (UVQ) [66] employs a fixed codebook generated using a large set of training vectors selected from different types of images. Input vectors are encoded using this codebook and labels are transmitted to the receiver. If the codebook size is N, then the bit rate is given by,

$$R = \frac{(\log_2 N)}{L} \quad bits/pixel \tag{2.41}$$

where L is the vector dimension.

To ensure good image fidelity, the codebook size must be large, which increases both the bit rate and the coding complexity. In addition, good coding performance may not be achieved for various types of input images. Recently, solutions which address these problems have been proposed. The codebook size can be decreased using techniques which exploit the image features. Examples include classified VQ, predictive VQ and finite state VQ. In classified VQ (CVQ) [67, 68], the training vectors are classified into a finite set of classes based on perceptually important features, such as, the edges. For each class, a separate codebook is generated. In the quantization process, a classifier determines the class of each input vector which are encoded using appropriate codebooks. Predictive VQ (PVQ) schemes employ a predictor followed by vector quantizer [65]. The input vector is predicted from the previously transmitted vectors, and the prediction error vector is quantized. For error vectors, a smaller size codebook can be employed. Another approach which reduces the bit rate and the search complexity is the finite state VQ (FSVQ) [69]. A FSVQ encoder consists of a finite set of states, a small subcodebook is associated with each state. The state of the encoder is determined from the previously encoded vectors and the input vector is quantized using the corresponding subcodebook.

2.4 Video Compression Schemes

A video signal contains the two types of basic correlations, namely, temporal (inter-frame) and spatial (intra-frame). Compression of the video signal is achieved by removing the temporal and spatial correlations using the inter-frame and intra-frame coding techniques, respectively [3]. Hence, a video compression scheme is a combination of the inter/intra-frame coding techniques. In this section, we review video compression techniques including motion compensated temporal prediction (MCP), hybrid DPCM/DCT coding and inter-frame VQ. Buffer control strategies which are important in constant (low) bit rate applications are also briefly reviewed.

2.4.1 Motion Estimation and Compensation

Motion estimation and compensation has become an integral part of low bit rate video compression schemes because of its ability to exploit the temporal correlation present in the video signal. Motion estimation techniques attempt to obtain the motion information for the objects in the frame. Several algorithms have been developed to calculate the motion information which can be broadly classified into two groups: pel-recursive and block-matching [26, 8].

Pel-recursive algorithms (PRA) use previously transmitted information to obtain a motion vector for every pixel, and are often called backward estimations. These algorithms do not require the transmission of motion information but recursively use the relative luminance change to find the motion information. Several pel-recursive techniques have been proposed which are reviewed in [26]. It has been found that the PRAs are susceptible to noise and compensation is often unsatisfactory. In addition, these algorithms need large amounts of computation.

Block-matching algorithms (BMA) require the transmission of motion information of the predetermined blocks in contrast to the pel-recursive algorithms, and are often

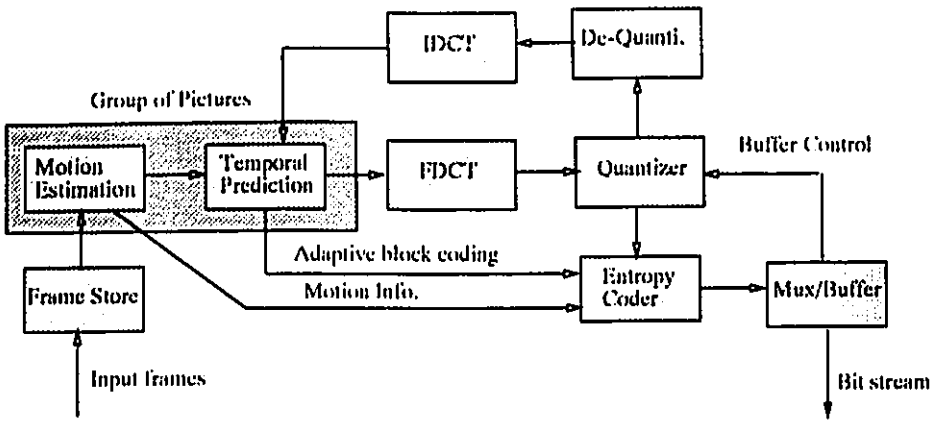

Figure 2.6: Hybrid encoder for video compression

called forward estimations. Here, it is assumed that all the pixels in a block undergo the same displacement. The mean square error (MSE) or the mean absolute difference (MAD) is used as a matching criterion. Exhaustive search over the whole search area gives a global minimum vector but it involves a higher computational complexity. Wang *et al* [26] have compared the performance of several fast searching techniques such as logarithmic or binary search [27], conjugate direction search, hierarchical search [28], etc. The block-matching techniques are widely used compared to the pel-recursive techniques because of the superior performance and lower computational complexity.

We note that the motion information of a frame obtained from the motion estimation can be used to predict the frame from the reference frames. This process is called the motion compensation. Thus, motion compensation attempts to reduce the frame differences and thereby achieve data compression.

2.4.2 Hybrid Transform Video Coding

A hybrid coding scheme which is a combination of Differential Pulse Code Modulation

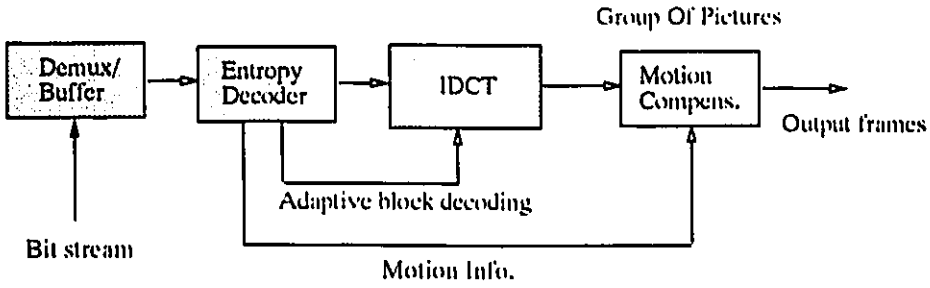

Figure 2.7: Hybrid decoder for video compression

(DPCM) and Discrete Cosine Transform (DCT) is most widely used for video compression. It exploits the temporal correlation using inter-frame DPCM and spatial correlation using intra-frame variable length DCT (VLDCT) [18]. Note that hybrid coding scheme is recommended as the source coding scheme in the MPEG and H.261 video compression standards. The block diagram of this scheme is shown in Fig. 2.6 which is explained as follows.

Motion Estimation and Temporal Prediction

Motion estimation is performed on the current frame using already coded frames. The motion information is transmitted to the receiver and used to predict or interpolate the current frame. The block-matching algorithm (BMA) discussed in section 2.4.1 is used to obtain the motion information. In BMA, smaller block size results in a higher coding gain but requires more bits to transmit the motion information. The choice of 16×16 blocks is the trade-off between coding gain and bit rate requirement. The motion-compensation blocks are also called macroblocks. Each of these macroblocks can be intra, forward-predicted, backward predicted or averaged as shown in the encoding flow diagram in Fig. 2.8. Adaptivity is achieved by properly selecting the encoding path for each macroblock. For example, four error macroblocks are

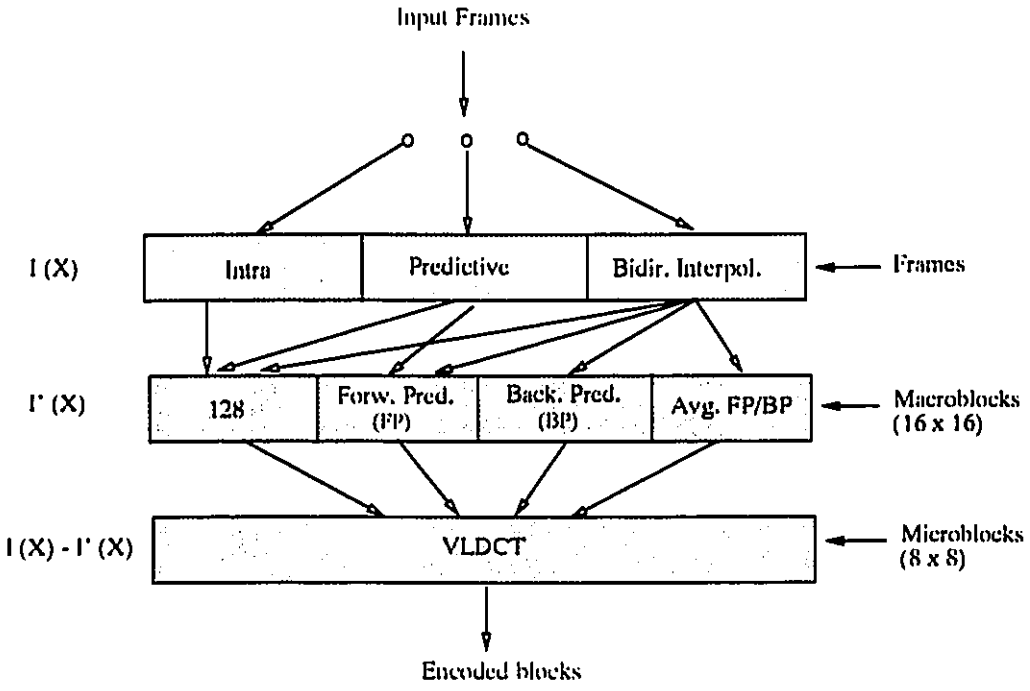

Figure 2.8: Encoding flow for adaptive coding

calculated for the B frames using the forward prediction, backward prediction, average of the forward and backward prediction and subtracting 128 from each pixel of the block and the macroblock with the smallest energy is selected for coding. For the I frames, macroblocks are always obtained by subtracting 128 from each pixel value. The motion vectors obtained for each macroblock are DPCM/Huffman coded whereas the error macroblocks are VLDCT coded as discussed below.

DCT Coding and Quantization

Variable length DCT (VLDCT) is an efficient technique to remove the spatial redundancy from the signal [43]. In a hybrid coder, VLDCT is performed on the error images on a block basis. A block size of 8 × 8 is found to be a good compromize be-

tween computational complexity and performance. DCT of a given block is obtained as,

$$F(u,v) = \frac{1}{4}C(u)C(v)\left[\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y) * \cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16}\right] \qquad (2.42)$$

where $C(u),\ C(v) = 1/\sqrt{2}$ *for* $u,\ v = 0$ *and* $C(u),\ C(v) = 1$ *otherwise.*

The signal is recovered by the inverse DCT operation defined as,

$$f(x,y) = \frac{1}{4}\left[\sum_{x=0}^{7}\sum_{y=0}^{7} C(u)C(v)F(u,v) * \cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16}\right] \qquad (2.43)$$

DCT converts the input signal into 64 unique 2D spatial frequencies. The coefficient with zero frequency is called the DC coefficient and remaining 63 are called the AC coefficients. Each of the 64 DCT coefficients is quantized using a predefined quantization table. The purpose of the quantization is to achieve further compression by removing the irrelevent information. Quantized coefficients are obtained by dividing each DCT coefficient by the quantizer step size, followed by rounding operation,

$$F^Q(u,v) = Round\left(\frac{F(u,v)}{Q(u,v)}\right). \qquad (2.44)$$

Note that the step size should be chosen such that the resulting degradation is not visually impairing. The intra-coded blocks are quantized differently than the non-intra coded blocks as shown in Fig. 2.9 because of the different frequency distributions. In each block, the DC coefficient is treated separately from the 63 AC coefficients because it contains a significant fraction of the total image energy. The DC coefficient is DPCM coded with respect to the DC coefficient of the previous block. Each AC coefficient is compared to a fixed threshold and the coefficients below the threshold value are set to zero. Finally, all the quantized coefficients are zig-zag scanned as shown in Fig. 2.10 in order to place the low frequency coefficients before the high frequency coefficients.

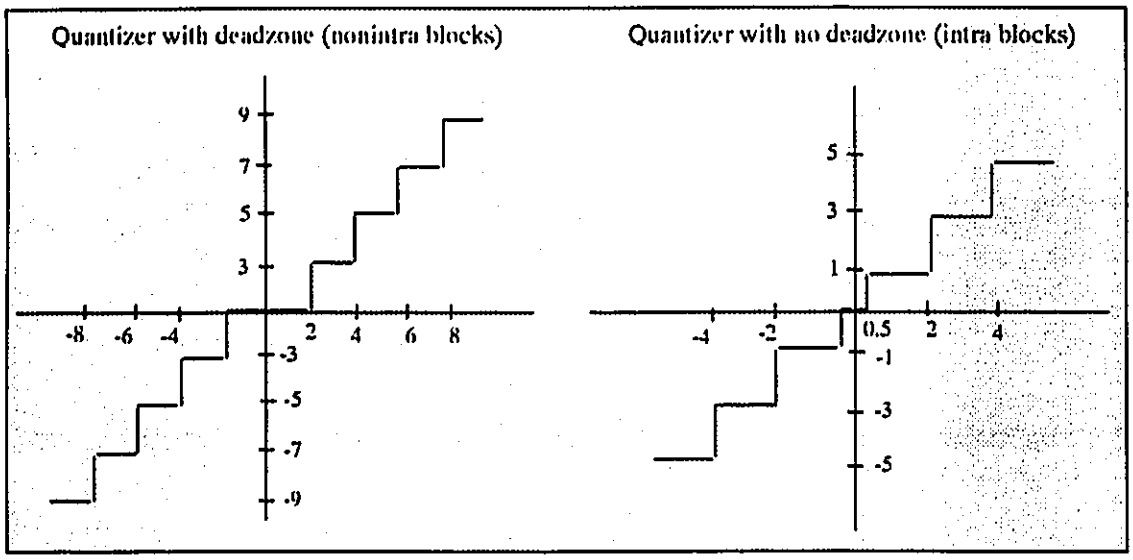

Figure 2.9: Quantizers for intra-coded and non-intra coded DCT blocks in the MPEG encoder

Entropy Coding

The final step in the encoder is entropy coding. This exploits the statistical redundancy and achieves further compression losslessly. Entropy coding can be considered as 2-step procedure. The first step converts the zig-zag sequence into an intermediate sequence of symbols. The second step converts the symbols into a bit stream.

A nonzero AC coefficient is represented as a combination of the Huffman and the run-length codes. Each run-length/nonzero coefficient combination is represented by a pair of symbols, (RUNLENGTH, SIZE) and (AMPLITUDE). RUNLENGTH is the number of consecutive zero-valued AC coefficients in the zig-zag sequence preceding the nonzero AC coefficient. SIZE is the number of bits used to represent AMPLITUDE by signed-integer Huffman coding. RUNLENGTH can represent zero-runs of length 0 to 15. Here, run-length of 15 specifies that the run of zero is larger than

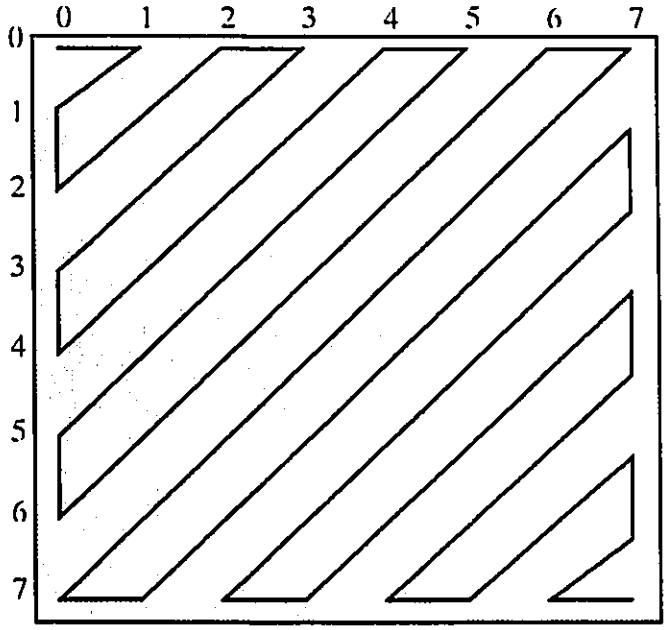

Figure 2.10: The zig-zag scan path of the DCT coefficients used in the MPEG encoder

15 and another run-length code is used for the rest of the run. The DC terms are represented by the two symbols, (SIZE) and (AMPLITUDE) and does not require the code of RUNLENGTH.

Once the coefficients are represented in the symbolic form, variable length codes are assigned. For the first symbol, Huffman tables are used. Two sets of Huffman tables are required, one for the DC terms and one for the AC terms. For the second symbol, variable length integer codes are used which are not Huffman codes. Based on these codes, a bit-stream is generated.

2.4.3 Inter-frame Vector Quantization

In this thesis, we employ UVQ for video compression and compare the performance with that of the hybrid transform coding. UVQ has been successfully used for video coding application [73, 77, 74]. Murakami *et al* [73] have proposed a vector quantizer for video signals where a fixed codebook is generated using a long training sequence of normalized (by mean and standard deviation) video signal. In the encoding step, after a 16-dimensional input vector has been normalized, the label, mean and standard deviation are transmitted to the receiver. Huguet *et al* [77] have extended the concept of 2-dimensional VQ to 3-dimensional VQ for video coding. In their technique, a video signal is divided into 3-dimensional blocks and a 3-dimensional codebook is employed to encode the input blocks. Nasrabadi *et al* [74] have proposed an inter-frame hierarchical address vector quantizer (IHAVQ) using quadtree segmentation. In IHAVQ, each frame is divided into the 7×7 blocks. A block matching algorithm (BMA) is used to estimate the motion of each block. A bit map is transmitted for motion/nomotion information. The difference between the current frame and the motion compensated frame is then segmented hierarchically into the different size blocks. The larger size blocks are replenished from the previous frame whereas a set of codebooks are used to encode the smaller size blocks. There are two problems with these algorithms. First of all, a limited size codebook is not sufficient to represent the different types of video sequences, whereas a large codebook increases the bit rate and complexity. Secondly, the coding performance may be degraded by the mismatch between the codebook and video sequences outside the training set.

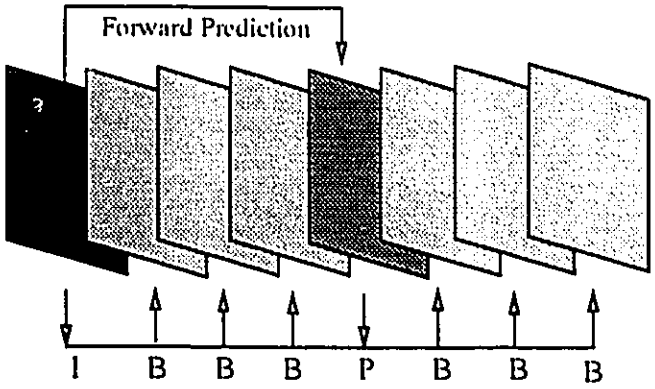

Figure 2.11: MPEG group of pictures (GOP)

2.5 Video Compression Standards

The Joint Photographic Experts Group (JPEG) and the Motion Pictures Experts Group (MPEG) of the International Standards Organization (ISO) have proposed standards for image and video compression known as the JPEG and MPEG-1 standards, respectively [15, 20]. In addition, the CCITT has recommended a standard for video compression called the H.261 at the bit rates of p $\times$ 64 where p is in the range 1 to 30 [19]. We briefly review the structure of the MPEG-1 video compression standard as it is used as a baseline for comparison.

2.5.1 MPEG-1 Video Compression Standard

In the MPEG-1 video compression standard, a group of pictures (GOP) approach is used instead of the frame by frame coding of video signals [20]. A GOP is a combination of one (typically) intra (I), one or two predicted (P) and the rest of bidirectional interpolated (B) pictures as shown in Fig. 2.11. The I pictures are useful in resetting the DPCM loop to avoid error accumulation and to provide random access points. The P pictures are coded with reference to already coded pictures and, they in

turn are used as the references to code B pictures. The B pictures achieve higher data compression and are motion compensation interpolated at the receiver from already transmitted I and P pictures. Hence, this is a two-level temporal pyramid approach [20]. We note that this pyramid does not switch adaptively to the spatial pyramid when there is a lower temporal correlation. For source coding, the hybrid coding scheme described in section 2.4.2 is recommended.

2.6 Buffer Control Strategies

Buffer control is very important in low bit rate environments where good picture quality is an essential requirement. Buffer control becomes difficult in the case of scene changes when more bits are required to code the difference frames. Several strategies have been proposed to solve this problem using a feedback signal which constantly monitors the buffer fullness [36, 37]. Here, when the buffer is full, the remaining information is coarsely quantized or skipped. This approach results in a variable quality in the different portions of the picture frames. Recently, a new approach has been proposed which uses the previous frame bit rate information to estimate the current frame bit rate and an iterative method to achieve the target bit rate [35].

A better approach to achieve uniform picture quality is a hierarchical coding scheme with proper bit allocation for the different levels. When the buffer is full, the final level can be coarsely quantized without sacrificing picture quality.

2.7 Required Features of a Video Codec

The basic requirements of the video coder are summarized below [20]:

- Random access to any video frame.
- Fast forward and reverse play-back in the encoded video.
- Feasibility of video editing.
- Robustness to the errors such as channel erros, coding errors, etc.
- Support different video input formats and resolutions.
- Possibility of low-cost, low-complexity implementation.

2.8 Video Signal Formats

The main requirement for a generic video codec is the ability to compress any given image format. A number of different formats are currently used in video coding applications. Table 2.1 gives a resume of some of the most widely used formats [81]. The Common Intermediate Format (CIF) and the Quarter Common Intermediate Format (QCIF) are generally used in video conference and video phone applications. Both formats are scanned in progressive mode. The color images are coded in terms of the luminance component Y and the chrominance components U and V. These components are related to R, G, and B components by the following matrix equation,

$$\begin{pmatrix} Y \\ U \\ V \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.146 & -0.288 & 0.434 \\ 0.617 & -0.517 & -0.100 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} \tag{2.45}$$

Because of the relatively small importance of the chrominance components, compared to the luminance component, they are subsampled by a factor of two in each direction. In addition, a higher distortion can be tolerated in the chrominance components without any perceptually visible artifacts.

The recommendation 601 of the CCIR committee proposes a format for digital TV known as CCIR-601 format [19]. This format is interlaced and the chrominance

Formats	lines (Y)	lines (U, V)	pels/line (Y)	pels/line (U, V)	scan	frames/field rate
CIF	288	144	360	180	1:1	29.97 Hz
QCIF	144	72	180	90	1:1	29.97 Hz
CCIR-601	480	240	720	360	1:2	25/50 Hz

Table 2.1: Characteristics of the commonly used formats in video coding (European standards)

components are subsampled by a factor of two in both the horizontal and vertical directions. In research environments, images of size $2^n \times 2^n$ coded at 8 bits/pixel are frequently used.

2.9 Summary

In this chapter, we have presented an overview of the basic techniques for image compression. We have first discussed some concepts of information theory. This is followed by a review of lossless and lossy coding techniques which include Huffman coding, run-length coding, predictive coding, transform coding and vector quantization. We have then described combinations of inter/intra-frame coding techniques for video compression. The techniques of motion compensated inter-frame prediction, hybrid transform coding and inter-frame vector quantization are detailed. This follows with a review of the proposed international video compression standards, particularly the MPEG-1 standard. Finally, the features of a video coder and different input video signal formats are summarized.

Chapter 3

Review of Pyramid Data Structures

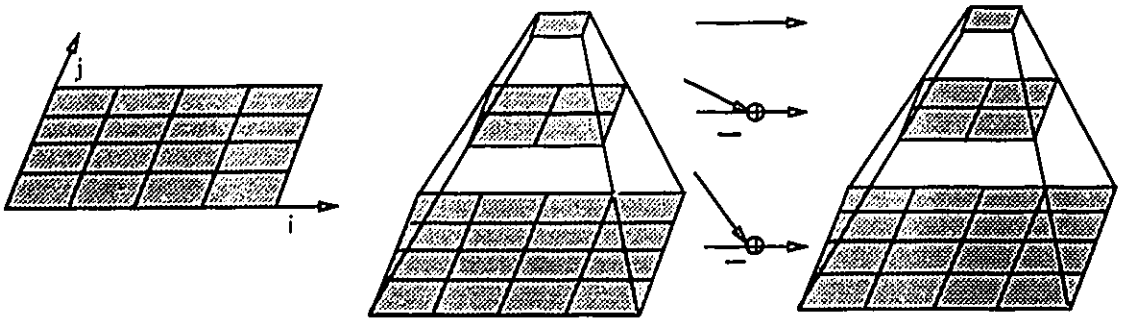

Figure 3.1: (a) 2D array of data (b) 2D pyramid (c) Difference pyramid

In this chapter, a review of pyramid data structures is presented. A pyramid data structure is simply a reorganization of the input data set with increasing resolutions from the top to bottom level of the pyramid. Note that the bottom level of the pyramid is the original data set. Pyramid data structures are widely used in the fields of image processing, computer graphics, geographical information systems, etc. A pyramid data structure is formed by successively obtaining reduced-resolution approximations of the data set. There are several methods to obtain the reduced-resolution approximations. Examples include Hadamard transform, low-pass filtering, averaging, summation, etc. [56]. A pyramid structure can be formed on 2D or 3D array of data resulting in a 2D or 3D pyramids. The details of the 2D and 3D pyramid data structures applicable to image/video compression is now presented.

3.1 2D Pyramid Data Structures

An image or a still video frame can be represented as a 2D data set as shown in Fig. 3.1. Hence, an image, X, can be written as,

$$X = \{X(i,j);\ i = 0,1,...M-1,\ \text{and}\ j = 0,1,...N-1\} \tag{3.1}$$

where i, j are the spatial indices, and $M \times N$ is the size of the image. Given an image X, a pyramid is defined as a sequence of matrices, $\{X_k\}$, such that, X_{k-1} is a reduced-resolution version of X_k. A 2D pyramid structure built using 2D data is shown in Fig. 3.1.

3.1.1 Mean Pyramid

For an image X as defined in Eqn. 3.1, the mean pyramid, $\{X_k\}$, is formed by successively averaging over 2×2 neighbouring nodes [56]. For a p-level pyramid, the steps of the algorithm can be written as follows:

1. *Base level:* Let $k = p$ and level k be the original data, $X_k = X$.

2. *Level $k-1$:* Level $k-1$ is obtained by calculating the mean for each spatially contiguous, non-overlapping block of 2×2 at level k,

$$X_{k-1}(i,j) = \frac{X_k(2i,2j) + X_k(2i,2j+1) + X_k(2i+1,2j) + X_k(2i+1,2j+1)}{4} \tag{3.2}$$

   where $i = 0, 1, ...M/2^{p-k+1} - 1$ and $j = 0, 1, ...N/2^{p-k+1} - 1$.

3. *Termination:* Let $k = k - 1$ and if $k \neq 0$, return to step 2; otherwise, stop.

The intermediate levels of the mean pyramid, $\{X_k\}$, are a set of reduced-resolution approximations of the image where the bottom level, X_n, is the original image and the top level, X_0, the mean value of the image.

There are a number of problems with the mean pyramid structure for image compression. First of all, the total number of nodes is,

$$\begin{aligned} \sum_{k=0}^{p} \frac{M \times N}{2^{p-k} \times 2^{p-k}} &= \frac{M \times N}{2^p \times 2^p} \sum_{k=0}^{p} 2^k \times 2^k &= \frac{M \times N}{2^p \times 2^p} \frac{2^{p+1} \times 2^{p+1} - 1}{2 \times 2 - 1} \\ &= M \times N \left[\frac{4}{3} - \frac{1}{3(2^p \times 2^p)}\right] &\approx \frac{4}{3}[M \times N] \end{aligned} \tag{3.3}$$

which is about 1/3 more than the total original data ($M \times N$) which results in data expansion. Secondly, extra bits are required to accurately record the node values, as precision of the data is changed. To solve the second problem, the node values are rounded off to the same resolution as the original data. The resulting pyramid is often called the *truncated pyramid* [56]. Here, step 2 (Eqn. 3.2) is replaced by the following step,

$$X^*_{k-1}(i,j) = \left[\frac{X_k(2i,2j) + X_k(2i,2j+1) + X_k(2i+1,2j) + X_k(2i+1,2j+1)}{4}\right] \tag{3.4}$$

where $i = 0, 1, ...M/2^{p-k+1} - 1$ and $j = 0, 1, ...N/2^{p-k+1} - 1$ and $[\alpha]$ is the truncation of $\alpha + 0.5$.

Finally, since this pyramid simply consists of a set of reduced-resolution approximations of the original data, there exists a high degree of correlation between the neighbouring nodes at each level and between the nodes and their children. The first order entropy of the mean pyramid, H_m (bits/node), is almost the same as the first order entropy of the original image, H_0 (bits/pixel),

$$H_m \approx H_0. \tag{3.5}$$

3.1.2 Sum Pyramid

The sum pyramid, $\{S_k\}$, is obtained by using the sum operation instead of the mean operation used in the mean pyramid. For a p-level pyramid, the steps of the algorithm can be written as follows:

1. *Base level:* Let $k = p$ and level k be the original data, $S_k = X$.

2. *Level $k-1$:* Level $k-1$ is obtained by calculating the sum for each spatially contiguous, non-overlapping block of 2×2 at level k,

$$S_{k-1}(i,j) = S_k(2i,2j) + S_k(2i,2j+1) + S_k(2i+1,2j) + S_k(2i+1,2j+1) \tag{3.6}$$

where $i = 0, 1, ...M/2^{p-k+1} - 1$ and $j = 0, 1, ...N/2^{p-k+1} - 1$

3. *Termination:* Let $k = k - 1$ and if $k \neq 0$, return to step 2; otherwise, stop.

At step 2, one of the 2×2 nodes can be dropped without losing any information, and original image X can be recovered by transmitting the retained node values in the sum pyramid. Hence, this pyramid is called the *reduced-sum pyramid* [56]. We note that the dropped value can be calculated from the neighbouring nodes and parents. The total number of nodes is equal to the number of pixels in X,

$$\begin{aligned} \frac{M\times N}{2^p\times 2^p} + \frac{3}{4}\sum_{k=1}^{p} \frac{M\times N}{2^{p-k}\times 2^{p-k}} &= \frac{M\times N}{2^p\times 2^p}\left[1 + \frac{3}{4}\sum_{k=1}^{p} 2^k \times 2^k\right] \\ &= \frac{M\times N}{2^p\times 2^p}\left[1 + \frac{3}{4}\times 2\times 2\times \frac{2^p\times 2^p - 1}{2\times 2 - 1}\right] = M \times N \end{aligned} \tag{3.7}$$

We note that the reduced-sum pyramid still has the problems of round off errors and high neighbouring node correlation.

3.1.3 Gaussian Pyramid

A Gaussian pyramid is a general approach to form a pyramid. In this pyramid, the image, X, is first successively convolved with a Gaussian-like weighting function, producing a set of low-pass filtered i.e. reduced-resolution approximations, $\{G_k\}$, called the *Gaussian pyramid.* For a p-level pyramid, the steps of the algorithm can be written as follows:

1. *Base level:* Let $k = p$ and level k be the original data, $X_k = X$.

2. *Level $k-1$:* The node values at level $k-1$ are calculated by convolving level k with a Gaussian-like weighting function $w(m, n)$,

$$G_{k-1}(i, j) = \sum_{m,n} w(m, n) G_k(2i + m, 2j + n) \tag{3.8}$$

where $i = 0, 1, ...M/2^{p-k+1} - 1$ and $j = 0, 1, ...N/2^{p-k+1} - 1$. Note that the resolution of level $k-1$ is decimated by 2.

3. *Termination:* Let $k = k - 1$ and if $k \neq 0$, return to step 2; otherwise, stop.

The Gaussian pyramid has a number of problems similar to the mean pyramid. First of all, extra bits are required to record exactly the node values in the Gaussian pyramid. This problem can be solved by truncating the node values of the Gaussian pyramid. Here, step 2 (Eqn. 3.8) is replaced by,

$$G^*_{k-1}(i,j) = \left[\sum_{m,n} w(m,n) G^*_k(2i+m, 2j+n)\right] \tag{3.9}$$

where $i = 0, 1, ...M/2^{p-k+1} - 1$ and $j = 0, 1, ...N/2^{p-k+1} - 1$ and $[\alpha]$ is the truncation of $\alpha + 0.5$.

Note that the bottom level of the truncated Gaussian pyramid, G^*_k, is still the original image, X. Secondly, the Gaussian pyramid has the same number of extra nodes as the mean pyramid (Eqn. 3.3). Finally, the Gaussian pyramid also has adjacent node and adjacent level correlations. Hence, this pyramid is not a suitable structure for data compression.

3.1.4 S-Transform Pyramid

The S-transform pyramid, $\{T_k\}$, addresses the problems of extra nodes and higher correlation at the same time. The basic operation in the S-transform pyramid is the successive application of the 2×2 Hadamard transform. For a p-level pyramid, the steps of the algorithm can be written as follows:

1. *Base level:* Let $k = p$ and level k be the original data, $S^*_k = X$.

2. *Level $k - 1$:*

(a) Level k undergoes a block Hadamard transform of size 2×2,

$$\begin{cases} T_k(2i,2j) & = \frac{1}{2}(S_k^*(2i,2j) + S_k^*(2i,2j+1) + S_k^*(2i+1,2j) + S_k^*(2i+1,2j+1)) \\ T_k(2i,2j+1) & = \frac{1}{2}(S_k^*(2i,2j) - S_k^*(2i,2j+1) + S_k^*(2i+1,2j) - S_k^*(2i+1,2j+1)) \\ T_k(2i+1,2j) & = \frac{1}{2}(S_k^*(2i,2j) + S_k^*(2i,2j+1) - S_k^*(2i+1,2j) - S_k^*(2i+1,2j+1)) \\ T_k(2i+1,2j+1) & = \frac{1}{2}(S_k^*(2i,2j) - S_k^*(2i,2j+1) - S_k^*(2i+1,2j) + S_k^*(2i+1,2j+1)) \end{cases} \tag{3.10}$$

where $i = 0, 1, ...M/2^{p-k+1} - 1$ and $j = 0, 1, ...N/2^{p-k+1} - 1$.

$T_k(2i, 2j)$ represents a sum coefficient, and $T_k(2i, 2j+1)$, $T_k(2i+1, 2j)$ and $T_k(2i+1, 2j+1)$ are the horizontal, vertical, and diagonal difference coefficients, respectively. Note that the sum coefficients are not transmitted with the difference coefficients.

(b) The sum coefficients are assembled to form a reduced-resolution approximation of the original image, S_{k-1}^*,

$$S_{k-1}^*(i,j) = T_k^*(2i,2j) \tag{3.11}$$

where $i = 0, 1, ...M/2^{p-k+1} - 1$ and $j = 0, 1, ...N/2^{p-k+1} - 1$.

3. *Termination:* Let $k = k - 1$ and if $k \neq 0$, return to step 2; otherwise, stop.

In Eqn. 3.10, one node is skipped in each 2×2 block, and hence, this is a reduced pyramid. To recover the original image, X, we need to transmit only the top level, S_0^*, and the difference coefficients. The inverse Hadamard transform is applied on the difference coefficients to reconstruct the original image. As the pyramid has difference nodes, the first order entropy of the pyramid, H_t (bits/node), is less than the entropy, H_0 (bits/pixel), of the original image,

$$H_t < H_0. \tag{3.12}$$

3.2 Difference Pyramids

It has been found that the first order entropy of the difference nodes is less than that of the original nodes in a pyramid. Hence, in data compression application, the difference pyramids are used. In this section, we review the difference and reduced difference pyramids derived from the mean pyramid. The difference pyramid for the Gaussian pyramid results in a Laplacian pyramid. We review the Laplacian pyramid, as it is widely used for image/video compression [60, 61, 54, 56].

3.2.1 Difference Pyramid

The difference pyramid, $\{D_k\}$, is formed by successively taking the difference between adjacent levels of the truncated mean pyramid. For a p-level pyramid, the steps of the algorithm can be written as follows:

1. *For level* $k = p, p-1, \ldots 2$: The level $k-1$ is interpolated by repeating the node values and subtracted from the corresponding nodes at level k,

$$\begin{cases} D_k(2i, 2j) & = X_k^*(2i, 2j) - X_{k-1}^*(i, j) \\ D_k(2i, 2j+1) & = X_k^*(2i, 2j+1) - X_{k-1}^*(i, j) \\ D_k(2i+1, 2j) & = X_k^*(2i+1, 2j) - X_{k-1}^*(i, j) \\ D_k(2i+1, 2j+1) & = X_k^*(2i+1, 2j+1) - X_{k-1}^*(i, j) \end{cases} \tag{3.13}$$

   where $i = 0, 1, \ldots M/2^{p-k+1} - 1$ and $j = 0, 1, \ldots N/2^{p-k+1} - 1$.

2. *For the top level,* $k = 1$:

$$D_k(i, j) = X_k^*(i, j). \tag{3.14}$$

The number of nodes in the difference pyramid is equal to the number of nodes in the truncated mean pyramid. The node values of the difference pyramid are concentrated

around zero and hence, the first order entropy of the difference pyramid, H_d, is much less than the first order entropy of the original image, H_0,

$$H_d < H_0. \tag{3.15}$$

In the difference pyramid, lossless coding of the signal is possible eventhough a lossy coder is employed at the upper levels. Here, the coding errors introduced at the upper levels are delivered to the lower levels and thus, the information lost is reprocessed. This property of the difference pyramid is known as *error-delivery* [56, 57].

3.2.2 Reduced Difference Pyramid

The reduced difference pyramid, $\{D_k^*\}$, is obtained from the truncated mean pyramid (Eqn. 3.4) by first taking the differences between the neighbouring nodes for each spatially contiguous, nonoverlapping block of size 2×2, at level k. In algorithmic form, the steps can be written as follows:

1. *For level* $k = p, p-1, \ldots 2$: The difference is taken between the neighbouring nodes in each 2×2 block at level k,

$$\begin{cases} D_k^*(2i, 2j) & = X_k^*(2i, 2j) - X_k^*(2i, 2j+1) \\ D_k^*(2i, 2j+1) & = X_k^*(2i, 2j+1) - X_k^*(2i+1, 2j+1) \\ D_k^*(2i+1, 2j+1) & = X_k^*(2i+1, 2j+1) - X_k^*(2i+1, 2j) \\ D_k^*(2i+1, 2j) & = X_k^*(2i+1, 2j) - X_k^*(2i, 2j) \end{cases} \tag{3.16}$$

   where $i = 0, 1, \ldots M/2^{p-k+1} - 1$ and $j = 0, 1, \ldots N/2^{p-k+1} - 1$.

   Here, one node can be discarded without losing any information to losslessly reconstruct the original data.

2. *For the top level,* $k = 1$:

$$D_k^*(i, j) = X_k^*(i, j). \tag{3.17}$$

In a reduced pyramid, three out of four nodes are retained and hence the total number of nodes is equal to the number of pixels in the image. It can be easily shown that the truncated mean pyramid can be losslessly reconstructed from the reduced pyramid. The reduced pyramid demonstrates the necessary features useful for data compression such as lower correlation, exact number of nodes, error delivery, etc. [58].

3.2.3 Laplacian Pyramid

The Laplacian pyramid, $\{L_k\}$, is formed by taking the differences between successive levels of the truncated Gaussian pyramid, $\{G_k^*\}$. In algorithmic form, the steps can be written as follows:

1. *For level* $k = p, p-1, \ldots 2$:

   (a) The Gaussian pyramid level $k-1$ is first interpolated using the Gaussian filter kernel,

   $$\hat{G}_k^*(2i, 2j) = \sum_{m,n} w(m,n) G_{k-1}^*(i-m, j-n) \tag{3.18}$$

   (b) The difference is taken between the interpolated level, $\hat{G}_k^*$, and the level k,

   $$L_k(2i, 2j) = G_k^*(2i, 2j) - \hat{G}_k^*(2i, 2j) \tag{3.19}$$

   where $i = 0, 1, \ldots M/2^{p-k+1} - 1$ and $j = 0, 1, \ldots N/2^{p-k+1} - 1$.

2. *For the top level,* $k = 1$:

   $$L_k(i, j) = G_k^*(i, j). \tag{3.20}$$

The Laplacian pyramid contains all the necessary information to losslessly reconstruct the original image. As the Laplacian pyramid consists of a set of decorrelated difference values, the entropy of the Laplacian pyramid is expected to be less than

that of the Gaussian pyramid. Therefore more efficient transmission can be achieved by encoding the Laplacian pyramid, instead of the Gaussian pyramid. We note that the Laplacian pyramid still has extra nodes similar to the Gaussian pyramid.

The Gaussian pyramid can be reconstructed from the Laplacian pyramid as follows.

1. *For the top level,* $k = 1$*:*

$$G_k^*(i,j) = L_k(i,j) \tag{3.21}$$

where $i = 0, 1, ...M/2^{p-k+1} - 1$ and $j = 0, 1, ...N/2^{p-k+1} - 1$.

2. *For level,* $k = 2, 3, \ldots p$*:*

(a) The previously reconstructed Gaussian pyramid level $k-1$ is first interpolated using the Gaussian filter kernel,

$$\hat{G}_k^*(2i, 2j) = \sum_{m,n} w(m,n) G_{k-1}^*(i-m, j-n) \tag{3.22}$$

where $i = 0, 1, ...M/2^{p-k+1} - 1$ and $j = 0, 1, ...N/2^{p-k+1} - 1$.

(b) The interpolated Gaussian level is added to the Laplacian pyramid level at k,

$$G_k^*(2i, 2j) = L_k(2i, 2j) + \hat{G}_k^*(2i, 2j) \tag{3.23}$$

where $i = 0, 1, ...M/2^{p-k+1} - 1$ and $j = 0, 1, ...N/2^{p-k+1} - 1$.

Note that the bottom level, L_0, is the original image.

3.3 3D Pyramid Data Structures

The video sequence is first divided into groups of L frames, each group of frames can then be represented as a finite 3D array,

$$X = \{X(i,j,l);\ i = 0, 1, ...M-1,\ j = 0, 1, ...N-1,\ \text{and}\ l = 0, 1, ...L-1\} \tag{3.24}$$

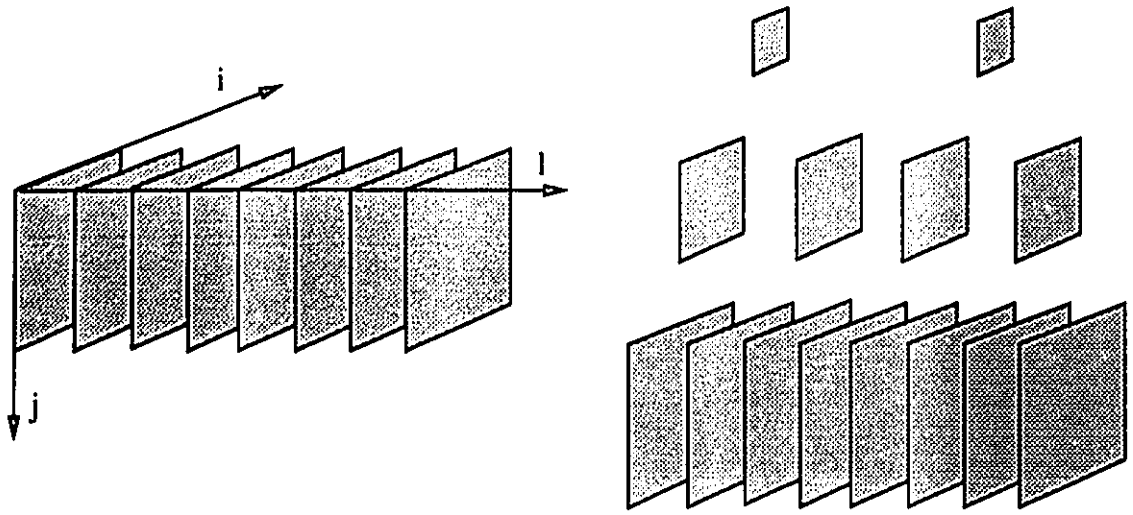

Figure 3.2: (a) 3D array of data (b) 3D spatio-temporal pyramid

where i, j are the spatial indices and l is the temporal index, as shown in Fig. 3.2. A 3D pyramid structure can be formed using this data by successively reducing the resolution of the data in both the temporal and spatial dimensions as shown in Fig. 3.2. These structures can exploit the temporal as well as spatial correlations present in the video signal. Recently, there is an increasing interest in 3D pyramid data structures for video compression [59, 60, 61]. In this section, we present a review of the algorithms to construct 3D pyramid data structures.

3.3.1 Spatio-temporal Pyramid

Uz *et al* [59] have proposed a spatio-temporal pyramid in which a group of frames is decimated in temporal and spatial domains to form the lower resolution level of the pyramid as shown in Fig. 3.2. In algorithmic form, the steps to form the spatio-temporal, $\{X_k^{ST}\}$, pyramid can be written as follows:

1. *Base level:* Let $k = p$ and level k be the original data, $X_k^{ST} = X$.

2. *Level* $k-1$*:*

   (a) *Temporal decimation:* First the level k is decimated in the temporal dimension of the signal,

$$X_{k-1}^{T}(i,j,l) = X_{k}^{ST}(i,j,2l) \tag{3.25}$$

   where $i = 0,1,\ldots M-1$, $j = 0,1,\ldots N-1$ and $l = 0,1,\ldots L/2^{p-k+1}-1$.

   (b) *Spatial decimation:* Temporally decimated level k is now decimated in spatial dimension giving the level $k-1$,

$$X_{k-1}^{ST}(i,j,l) = X_{k-1}^{T}(2i,2j,l) \tag{3.26}$$

   where $i = 0,1,\ldots M/2^{p-k+1}-1$, $j = 0,1,\ldots N/2^{p-k+1}-1$
   and $l = 0,1,\ldots L/2^{p-k+1}-1$.

3. *Termination:* Let $k = k-1$ and if $k \neq 0$, return to step 2; otherwise, stop.

The intermediate levels of the spatio-temporal pyramid, $\{X_k^{ST}\}$, are a set of reduced spatial and temporal resolution approximations of the video sequence, where the bottom level is the original sequence.

At the decoder, the signal is first spatially interpolated followed by motion compensated temporal interpolation. We note that in motion compensation, the motion information about the objects in the video frames is used for temporal interpolation. There are a number of problems in this pyramid. First of all, the pyramid has extra nodes given by,

$$\begin{aligned}\sum_{k=0}^{p} \frac{M\times N\times L}{2^{p-k}\times 2^{p-k}\times 2^{p-k}} &= \frac{M\times N\times L}{2^p\times 2^p\times 2^p}\sum_{k=0}^{p} 2^k\times 2^k\times 2^k &&= \frac{M\times N\times L}{2^p\times 2^p\times 2^p}\frac{2^{p+1}\times 2^{p+1}\times 2^{p+1}-1}{2\times 2\times 2-1}\\ &= M\times N\times L\left[\frac{8}{7}-\frac{1}{7(2^p\times 2^p\times 2^p)}\right] &&\approx \frac{8}{7}[M\times N\times L]\end{aligned} \tag{3.27}$$

which is about 14 % higher than the original data. Comparing Eqns. 3.3, 3.27, it is obvious that the number of extra nodes reduces as the dimension of the pyramid

increases. Secondly, this pyramid is not adaptive to motion and it can fail in the event of fast motion and scene changes. A better approach is to select either the temporal or spatial decimation based on the spatial and temporal correlation present in the video sequence. Finally, there exists a high degree of temporal and spatial correlation in the pyramid. The spatial correlation is between adjacent nodes and temporal correlation is between adjacent frames of a pyramid level. To address this problem, a difference pyramid is formed by taking the differences between successive levels of the spatio-temporal pyramid. In algorithmic form, the steps can be written as follows:

1. *For level* $k = p, p-1, \ldots 2$*:*

   (a) The level $k-1$ is spatially interpolated which is followed by motion compensated temporal interpolation,

$$\begin{aligned}
&\hat{X}^{ST}_{k-1}(2i,2j,2l) = \hat{X}^{ST}_{k-1}(2i+1,2j,2l) = \hat{X}^{ST}_{k-1}(2i,2j+1,2l) = \\
&\hat{X}^{ST}_{k-1}(2i+1,2j+1,2l) = \hat{X}^{ST}_{k-1}(2i,2j,2l+1) = \hat{X}^{ST}_{k-1}(2i+1,2j,2l+1) = \\
&\hat{X}^{ST}_{k-1}(2i,2j+1,2l+1) = \hat{X}^{ST}_{k-1}(2i+1,2j+1,2l+1) = X^{ST}_{k}(i,j,l)
\end{aligned} \tag{3.28}$$

   where $i = 0, 1, \ldots M/2^{p-k+1} - 1, \quad j = 0, 1, \ldots N/2^{p-k+1} - 1$

   and $l = 0, 1, \ldots L/2^{p-k+1} - 1$.

   (b) Interpolated level $k-1$, $\hat{X}^{ST}_{k-1}$, is subtracted from the corresponding nodes at level k,

$$\begin{cases}
D_k(2i,2j,2l) & = X^{ST}_k(2i,2j,2l) - \hat{X}^{ST}_{k-1}(2i,2j,2l) \\
D_k(2i,2j+1,2l) & = X^{ST}_k(2i,2j+1,2l) - \hat{X}^{ST}_{k-1}(2i,2j+1,2l) \\
D_k(2i+1,2j,2l) & = X^{ST}_k(2i+1,2j,2l) - \hat{X}^{ST}_{k-1}(2i+1,2j,2l) \\
D_k(2i+1,2j+1,2l) & = X^{ST}_k(2i+1,2j+1,2l) - \hat{X}^{ST}_{k-1}(2i+1,2j+1,2l) \\
D_k(2i,2j,2l+1) & = X^{ST}_k(2i,2j,2l+1) - \hat{X}^{ST}_{k-1}(2i,2j,2l+1) \\
D_k(2i,2j+1,2l+1) & = X^{ST}_k(2i,2j+1,2l+1) - \hat{X}^{ST}_{k-1}(2i,2j+1,2l+1) \\
D_k(2i+1,2j,2l+1) & = X^{ST}_k(2i+1,2j,2l+1) - \hat{X}^{ST}_{k-1}(2i+1,2j,2l+1) \\
D_k(2i+1,2j+1,2l+1) & = X^{ST}_k(2i+1,2j+1,2l+1) - \hat{X}^{ST}_{k-1}(2i+1,2j+1,2l+1)
\end{cases} \tag{3.29}$$

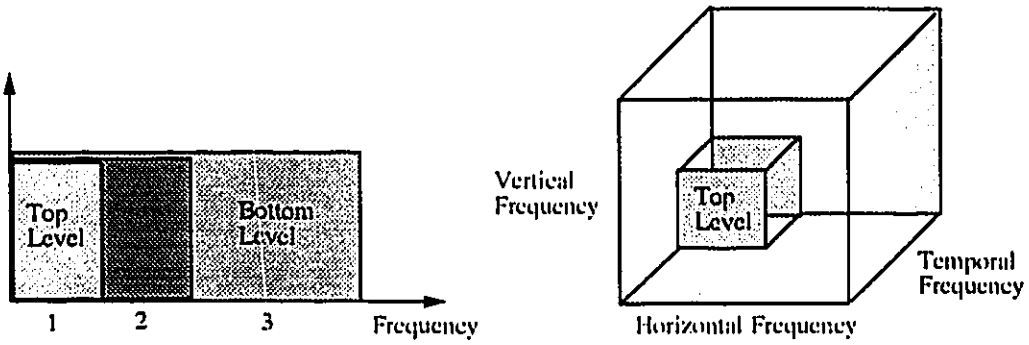

Figure 3.3: Logarithmic frequency division in pyramid (a) One dimensional pyramid with three levels (b) Three dimensional pyramid with two levels

where $i = 0, 1, ...M/2^{p-k+1} - 1, \quad j = 0, 1, ...N/2^{p-k+1} - 1$

and $l = 0, 1, ...L/2^{p-k+1} - 1.$

2. *For the top level,* $k = 1$*:*

$$D_k(i, j, l) = X_k^{ST}(i, j, l). \tag{3.30}$$

The nodes of the difference pyramids are highly decorrelated and error-delivery is possible in the pyramid. The number of nodes in the difference pyramid is the same as that of the spatio-temporal pyramid. The frequency division of a 3D pyramid is shown in Fig. 3.3 and is compared with that of a 1D pyramid.

3.3.2 3D Laplacian Pyramid

A video sequence is divided into individual frames and 2D Laplacian pyramid can be formed on each frame. To remove the temporal correlation between the frames, the prediction error frames can be used instead of the original frames or a 3D Laplacian pyramid can be constructed using 3D data directly.

Stiller *et al* [60] have used the Laplacian pyramid to code the motion compensated prediction error images. In algorithmic form, the steps can be written as follows:

1. *Base level:*

   (a) Find the predicted frame, $\hat{X}_k$, of the current frame X_k, and subtract it giving the prediction error frame,

   $$\hat{D}_k(i,j,2l) = X_k(i,j,2l) - \hat{X}_k(i,j,2l) \tag{3.31}$$

   where $i = 0,1,\ldots M-1$, $j = 0,1,\ldots N-1$ and $l = 0,1,\ldots L/2^{p-k+1} - 1$.

   (b) Let $k = p$ and level k be the prediction error frames, $L_k = \hat{D}_k$.

2. *For Level $k-1$:* Form the 2D Laplacian pyramid using Eqn. 3.19.

3. *Termination:* Let $k = k-1$ and if $k \neq 0$, return to step 2; otherwise, stop.

Although the coding results are promising, we have found that the pyramid increases the first order entropy of the error images and hence, results in data expansion. The reason for the data expansion is that the error images have very low spatial correlation coefficient (~ 0.3) compared to the original images (0.9-0.95) [33]. The bits are assigned to different parts of the error frames in the pyramid based on the prediction error energy.

Sallent *et al* [61] have employed a 3D adaptive Laplacian pyramid to code the video signal. The pyramid adapts to temporal vs. spatial correlation by selecting proper filtering matrices. The algorithmic steps can be written as follows:

1. *At the Base level $k = p$:* The original signal forms the base of the Gaussian pyramid, $G_k^* = X$.

2. *For level $k = p, p-1, \ldots 2$:*

(a) The low pass filtered level $k-1$ is first obtained using the Gaussian filter kernel, w,

$$G^*_{k-1}(2i,2j,2l) = \sum_{m,n,o} w(m,n,o) G^*_k(M_k(i+m, j+n, l+o)) \qquad (3.32)$$

where $i = 0,1,...M/2^{p-k+1}-1, \quad j = 0,1,...N/2^{p-k+1}-1$
and $l = 0,1,...L/2^{p-k+1}-1$. The M_k is the adaptive decimation matrix.

(b) The Gaussian pyramid level $k-1$ is then interpolated using the Gaussian filter kernel,

$$\hat{G}^*_k(2i,2j,2l) = (det(M_k)) \times \sum_{m,n,o} w(m,n,o) G^*_{k-1}(M_k^{-1}(i-m, j-n, l-o)) \qquad (3.33)$$

where $i = 0,1,...M/2^{p-k+1}-1, \quad j = 0,1,...N/2^{p-k+1}-1$
and $l = 0,1,...L/2^{p-k+1}-1$ and $(det(M_k))$ is the determinant of the matrix M_k.

(c) The difference is taken between the interpolated level, $\hat{G}^*_k$, and the level k,

$$L_k(2i,2j,2l) = G^*_k(2i,2j,2l) - \hat{G}^*_k(2i,2j,2l) \qquad (3.34)$$

where $i = 0,1,...M/2^{p-k+1}-1, \quad j = 0,1,...N/2^{p-k+1}-1$
and $l = 0,1,...L/2^{p-k+1}-1$.

3. *For the top level, $k = 1$:*

$$L_k(i,j,l) = G^*_k(i,j,l). \qquad (3.35)$$

The filtering matrices are selected such that the bandpass filtered version of the level has a Laplacian distribution. We note that in this pyramid, motion compensation was not employed.

3.4 Summary

In this chapter, we have presented a review of 2D and 3D pyramid data structues from the perspectives of data compression. First, the algorithms to form the 2D pyramids such as the mean pyramid, sum pyramid, S-transform pyramid and Gaussian pyramid are discussed. Difference pyramids are used for data compression because of their ability to reduce the first order entropy. Algorithms to construct the difference pyramid, reduced difference pyramid and Laplacian pyramid are then detailed. This follows with the 3D pyramids including the spatio-temporal pyramid and 3D Laplacian pyramids.

Chapter 4

3D Temporal/Spatial Pyramids for Video Compression

In this chapter, we propose algorithms to construct the temporal/spatial pyramid data structures for video compression. We recall from chapter 3 that a reduced-resolution approximation of a signal can be obtained by low-pass filtering the signal in either the temporal or spatial domains. After low-pass filtering, the signal is usually decimated by two or four. We use a subsampling filter that skips alternate samples in the temporal/spatial dimensions. We call this pyramid the *sample pyramid* as the sample values are not changed from one level to the next level. We show that this pyramid allows us to incorporate adaptive non-causal predictions. In addition, this pyramid is reduced and does not require the transmission of any overhead information. Performance of different configurations of the temporal/spatial pyramids is evaluated in terms of the first order entropy. Based on this study, we propose an efficient 3D adaptive temporal/spatial pyramid which achieves the lowest entropy.

4.1 Sample Pyramids

A 3D pyramid data structure can be formed by successively decimating the 3D array, X, in either the temporal or spatial dimension. We refer to these operations as *temporal decimation* and *spatial decimation*, respectively. A pyramid structure obtained from temporal decimations is called a *temporal pyramid* while a pyramid structure obtained from spatial decimations is called a *spatial pyramid*. In the following subsections, we present methods to form the temporal and spatial pyramids.

4.1.1 Temporal Sample Pyramid

The temporal sample pyramid, $X^T = \{X_k^T\}$, is formed by successively contracting the signal in the temporal dimension. For a p-level pyramid, the steps of the algorithm can be written as follows:

1. *Base level:* Let $k = p$ and level k be the original sequence, $X_k^T = X$.

2. *Level $k-1$:* Level $k-1$ is obtained by decimating the level k by 2 in temporal dimension, i.e.,

$$X_{k-1}^T(i,j,l) = X_k^T(i,j,2l) \tag{4.1}$$

where $i = 0,1,...M-1$, $j = 0,1,...N-1$, and $l = 0,1,...L/2^{p-k+1}-1$.

3. *Termination:* Let $k = k-1$ and if $k \neq 1$, return to step 2; otherwise, stop.

The original signal forms the base of the pyramid and intermediate levels, $\{X_k^T\}$, are reduced temporal resolution approximations of the original signal. Since level $k-1$ contains half of the frames of level k, a p-level pyramid will have

$$L \times (1 + \frac{1}{2} + ... + \frac{1}{2^{p-1}}) = L \times 2 \times (1 - \frac{1}{2^p}) \tag{4.2}$$

frames, or $(1 - \frac{1}{2^{p-1}})$ % more frames than the original video signal. For example, for $p = 4$, the pyramid has 87.5% more frames than the original video sequence. Furthermore, due to the temporal correlation in a video signal, there exists a high degree of correlation between the adjacent levels of the pyramid.

4.1.2 Spatial Sample Pyramid

The spatial sample pyramid, $X^S = \{X_k^S\}$, is formed by successively contracting the spatial dimensions of the signal. For a p-level pyramid, the steps of the algorithm can be written as follows:

1. *Base level:* Let $k = p$ and level k be the original sequence, $X_k^S = X$.

2. *Level $k-1$:* Level $k-1$ is obtained by decimating the level k by 2 in both the horizontal and vertical dimensions, i.e.,

$$X_{k-1}^S(i,j,l) = X_k^S(2i,2j,l) \tag{4.3}$$

where $i = 0, 1, 2, ...M/2^{p-k+1} - 1$, $j = 0, 1, 2, ...N/2^{p-k+1} - 1$, and $l = 0, 1, ...L-1$.

3. *Termination:* Let $k = k - 1$ and if $k \neq 1$, return to step 2; otherwise, stop.

The original signal forms the base of the pyramid and intermediate levels, $\{X_k^S\}$, are reduced spatial resolution approximations of the original signal. Eqn. 4.3 indicates that level $k-1$ has 1/4th the nodes of level k. The total number of nodes for a p-level pyramid is therefore equal to,

$$M \times N \times (1 + \frac{1}{2 \times 2} + ... + \frac{1}{2^{p-1} \times 2^{p-1}}) = M \times N \times \frac{4}{3} \times (1 - \frac{1}{2^p \times 2^p}) \quad (4.4)$$

which is $\frac{1}{3} \times (1 - \frac{1}{2^{p-1} \times 2^{p-1}})$ % more than the number of pixels in the original video signal. For p=4, the pyramid has about 33% more nodes than the original video sequence. The pyramid demonstrates a high degree of spatial correlation between the neighbouring nodes in the pyramid.

4.2 Difference Pyramids

The temporal and spatial sample pyramids are not suitable structures for coding purposes because of the data expansion and the high degree of correlations in both the temporal and spatial domains. To address these problems, the corresponding temporal and spatial difference pyramids are obtained by taking differences between successive levels of the temporal and spatial sample pyramids. Both pyramids have the same number of nodes as pixels in the original video signal and are highly decorrelated.

4.2.1 Temporal Difference Pyramid

The temporal difference pyramid, $D^T = \{D_k^T\}$, is obtained by taking differences between two adjacent levels of the temporal sample pyramid. In algorithmic form, it

can be written as follows:

1. *For level* $k = p, p-1, ...2$: Level $k-1$ is temporally interpolated by repeating the frames, and subtracted from the corresponding frames at level k,

$$\begin{cases} D_k^T(i,j,2l) & = X_k^T(i,j,2l) - X_{k-1}^T(i,j,l) \\ D_k^T(i,j,2l+1) & = X_k^T(i,j,2l+1) - X_{k-1}^T(i,j,l) \end{cases} \tag{4.5}$$

   where $i = 0, 1, ...M-1$, $j = 0, 1, ...N-1$, and $l = 0, 1, ...L/2^{p-k+1} - 1$.

   Using Eqn. 4.1, the difference for the even number frames,

$$D_k^T(i,j,2l) = 0 \tag{4.6}$$

   and hence, $D_k^T(i,j,2l)$ can be discarded from the difference pyramid without losing any information.

2. *For the top level* $k = 1$*:*

$$D_k^T(i,j,l) = X_k^T(i,j,l). \tag{4.7}$$

As even number frames are discarded, the temporal difference pyramid has the same number of frames as the original signal as shown in Fig. 4.1. Therefore, the pyramid is reduced. Furthermore, the temporal correlation between adjacent levels is decreased, since the pyramid consists of the temporal difference frames.

The temporal sample pyramid can be reconstructed from the temporal difference pyramid starting from the top level, $k = 1$, as follows:

1. *For the top level* $k = 1$*:*

$$X_k^T(i,j,l) = D_k^T(i,j,l). \tag{4.8}$$

2. *For level* $k = 2, 3, ...p$*:* The frames of level $k-1$ are temporally interpolated by repeating the frames and added to the corresponding difference frames at level

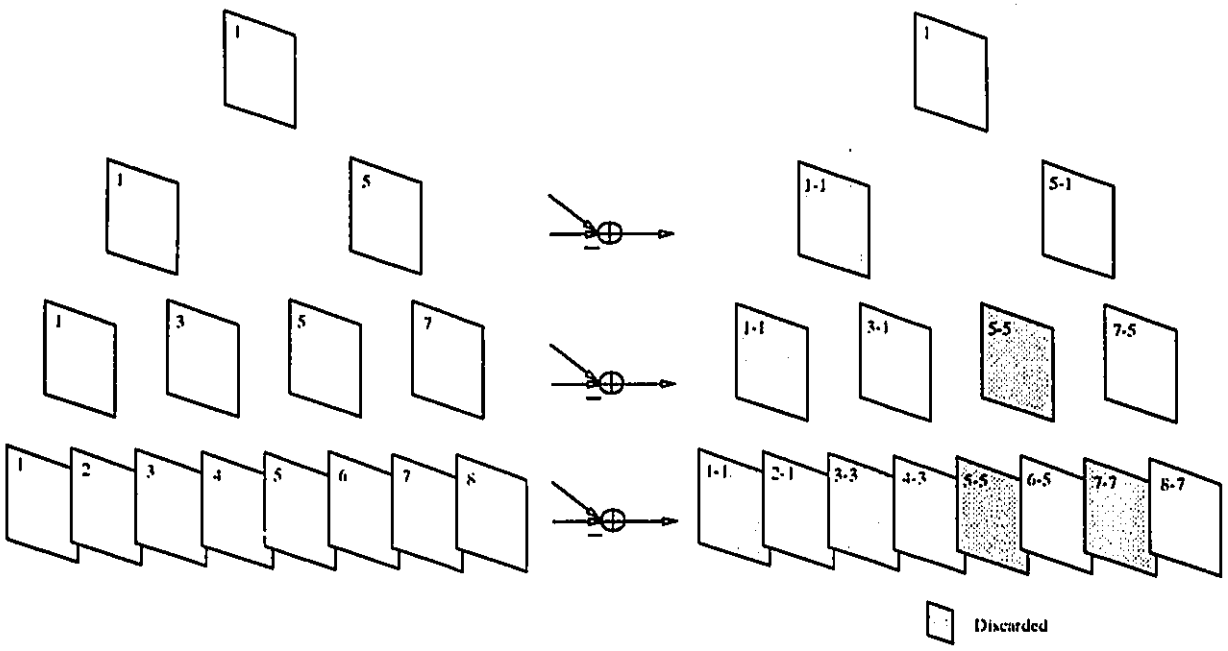

Figure 4.1: (a) Temporal sample pyramid (b) Temporal difference pyramid

k,

$$\begin{cases} X_k^T(i,j,2l) & = X_{k-1}^T(i,j,l) + 0 \\ X_k^T(i,j,2l+1) & = X_{k-1}^T(i,j,l) + D_k^T(i,j,2l+1) \end{cases} \tag{4.9}$$

where $i = 0, 1, ...M-1$, $j = 0, 1, ...N-1$, and $l = 0, 1, ...L/2^{p-k+1} - 1$. This is simply the inverse operation of the temporal sample pyramid formation.

Note that the base level, X_p^T, is the original video sequence, X, i.e., $X_p^T = X$.

4.2.2 Spatial Difference Pyramid

The spatial difference pyramid, $D^S = \{D_k^S\}$, can be obtained by taking differences between two adjacent levels of the spatial sample pyramid. In algorithmic form, it can be written as follows:

1. *For level $k = p, p-1, ...2$:* Level $k-1$ is spatially interpolated by repeating the nodes both horizontally and vertically, and subtracted from the corresponding nodes at level k,

$$\begin{cases} D_k^S(2i, 2j, l) & = X_k^S(2i, 2j, l) - X_{k-1}^S(i, j, l) \\ D_k^S(2i, 2j+1, l) & = X_k^S(2i, 2j+1, l) - X_{k-1}^S(i, j, l) \\ D_k^S(2i+1, 2j, l) & = X_k^S(2i+1, 2j, l) - X_{k-1}^S(i, j, l) \\ D_k^S(2i+1, 2j+1, l) & = X_k^S(2i+1, 2j+1, l) - X_{k-1}^S(i, j, l) \end{cases} \tag{4.10}$$

where $i = 0, 1, ...M/2^{p-k+1} - 1$, $j = 0, 1, ...N/2^{p-k+1} - 1$, and $l = 0, 1, ...L-1$.

Using Eqn. 4.3, we can obtain,

$$D_k^S(2i, 2j, l) = 0 \tag{4.11}$$

and hence, $D_k(2i, 2j, l)$ can be discarded from the difference pyramid without losing any information.

2. *For the top level $k = 1$:*

$$D_k^S(i, j, l) = X_k^S(i, j, l). \tag{4.12}$$

As each level discards 1/4th of its nodes, the total number of nodes is now equal to the number of pixels in the original video sequence as shown in Fig. 4.2. The spatial difference pyramid consists of the spatial differences and therefore has less spatial correlation.

The spatial sample pyramid can be reconstructed from the spatial difference pyramid starting from the top level, $k = 1$, as follows:

1. *For the top level $k = 1$:*

$$X_k^S(i, j, l) = D_k^S(i, j, l). \tag{4.13}$$

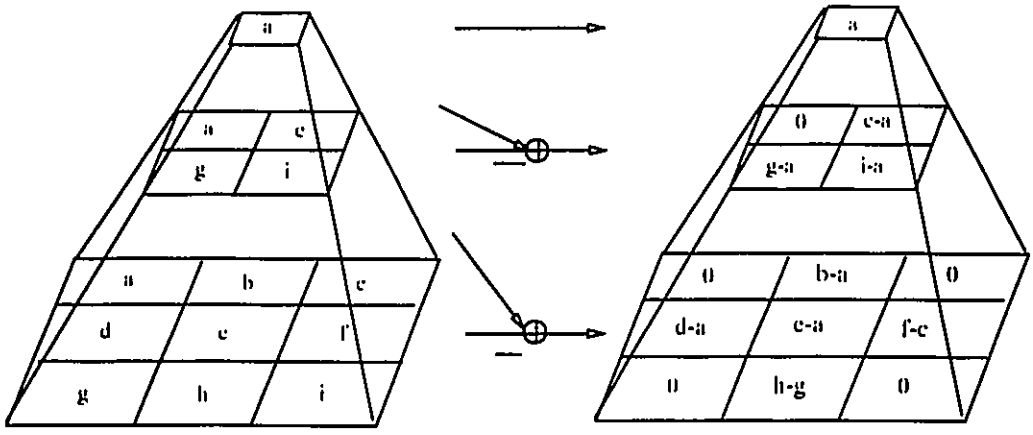

Figure 4.2: (a) Spatial sample pyramid (b) Spatial difference pyramid

2. *For level* $k = 2, 3, ...p$: The nodes of level $k-1$ are spatially zero-order interpolated and added to the corresponding difference nodes at level k,

$$\begin{cases} X_k^S(2i, 2j, l) & = X_{k-1}^S(i, j, l) + 0 \\ X_k^S(2i, 2j+1, l) & = X_{k-1}^S(i, j, l) + D_k^S(2i, 2j+1, l) \\ X_k^S(2i+1, 2j, l) & = X_{k-1}^S(i, j, l) + D_k^S(2i+1, 2j, l) \\ X_k^S(2i+1, 2j+1, l) & = X_{k-1}^S(i, j, l) + D_k^S(2i+1, 2j+1, l) \end{cases} \tag{4.14}$$

where $i = 0, 1, ...M/2^{p-k+1} - 1$, $j = 0, 1, ...N/2^{p-k+1} - 1$, and $l = 0, 1, ...L-1$.

The base level, X_p^S, is the original video sequence, X, i.e., $X_p^S = X$.

4.3 Prediction Difference Pyramids

The difference pyramids remove the temporal and spatial correlation by taking the difference between the adjacent levels of the sample pyramids where the upper levels are (zero-order) interpolated before subtracting from the lower level, as shown

in Eqn. 4.5 and Eqn. 4.10. To further reduce the correlations, the pyramid can be incorporated with non-causal prediction. Specifically, the missing frames in the temporal sample pyramid and the missing nodes in the spatial sample pyramid are, respectively, interpolated by using bidirectional motion estimation and compensation, and higher-order spatial prediction techniques. The resulting difference pyramid is called the *prediction difference pyramid*. The temporal pyramid results in temporal prediction difference pyramid and the spatial pyramid results in spatial prediction difference pyramid.

4.3.1 Temporal Prediction Difference Pyramid

The temporal prediction difference pyramid, $E^T = \{E_k^T\}$, can be formed by taking differences between the temporal sample level and its predicted level. In algorithmic form, it can be written as follows:

1. *For level* $k = p, p-1, ...2$:

   (a) The missing frames of level $k-1$ is temporally interpolated by bidirectional motion estimation and compensation, as shown in Fig. 4.3, and the interpolated level $k-1$ is used as the prediction for level k,

   $$\begin{cases} \hat{X}_k^T(i,j,2l) & = X_k^T(i,j,l) \\ \hat{X}_k^T(i,j,2l+1) & = predict\ (X_{k-1}^T(i,j,l), X_{k-1}^T(i,j,l+1)) \end{cases} \tag{4.15}$$

   where $i = 0,1,...M-1$, $j = 0,1,...N-1$, and $l = 0,1,...L/2^{p-k+1}-1$, and $predict()$ is the bidirectional motion estimation and compensation operation.

   (b) The temporal predictive difference is then obtained by,

   $$\begin{cases} E_k^T(i,j,2l) & = X_k^T(i,j,2l) - \hat{X}_k^T(i,j,2l) \\ E_k^T(i,j,2l+1) & = X_k^T(i,j,2l+1) - \hat{X}_k^T(i,j,2l+1) \end{cases} \tag{4.16}$$

where $i = 0, 1, ...M-1$, $j = 0, 1, ...N-1$, and $l = 0, 1, ...L/2^{p-k+1} - 1$.

It is clear from Eqn. 4.1 and Eqn. 4.15 that the differences for the even number frames are zero, i.e.,

$$E_k^T(i, j, 2l) = 0 \tag{4.17}$$

and hence, $E_k^T(i, j, 2l)$ can be discarded from the prediction difference pyramid without losing any information.

2. *For the top level $k = 1$:*

$$E_k^T(i, j, l) = X_k^T(i, j, l). \tag{4.18}$$

As even number frames are discarded, the temporal prediction difference pyramid has the same number of frames as the original signal as shown in Fig. 4.3. Therefore, the pyramid is reduced. Furthermore, since the prediction pyramid exploits the motion information, it usually demonstrates less temporal correlation than the difference pyramid. However, the motion vectors are required to reconstruct the temporal sample pyramid.

The temporal sample pyramid can be reconstructed from the temporal prediction difference pyramid starting from the top level, $k = 1$, as follows:

1. *For the top level $k = 1$:*

$$X_k^T(i, j, l) = E_k^T(i, j, l). \tag{4.19}$$

2. *For level $k = 2, 3, ...p$:* The missing frames of level $k-1$ are temporally interpolated using the motion vectors as given in Eqn. 4.15, and the interpolated level $k-1$ is then added to the corresponding difference frames at level k,

$$\begin{cases} X_k^T(i, j, 2l) & = \hat{X}_k^T(i, j, 2l) + 0 \\ X_k^T(i, j, 2l+1) & = \hat{X}_k^T(i, j, 2l+1) + E_k^T(i, j, 2l+1) \end{cases} \tag{4.20}$$

where $i = 0, 1, ...M-1$, $j = 0, 1, ...N-1$, and $l = 0, 1, ...L/2^{p-k+1} - 1$.

The base level, X_p^T, is the original video sequence, X, i.e., $X_p^T = X$.

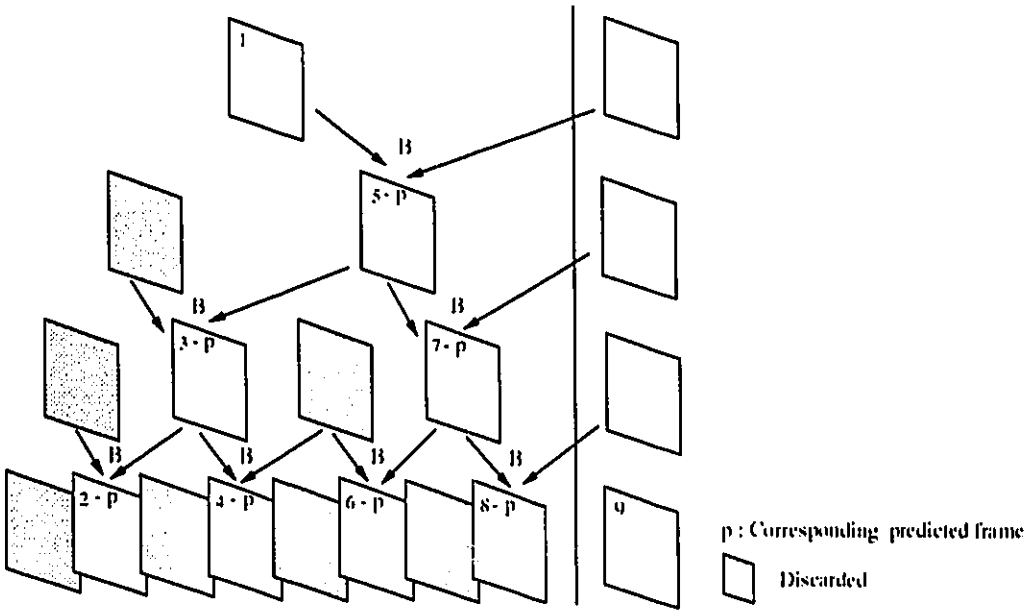

Figure 4.3: Temporal prediction difference pyramid

4.3.2 Spatial Prediction Difference Pyramid

The spatial prediction difference pyramid, $E^S = \{E_k^S\}$, can be obtained as follows:

1. *For level* $k = p, p-1, ...2$*:*

   (a) Level $k-1$ is spatially interpolated by using linear prediction both horizontally and vertically, or in other words, filling the missing nodes by averaging the neighbouring nodes, i.e.,

$$\begin{cases} \hat{X}_{k-1}^S(2i, 2j, l) & = X_{k-1}^S(i, j, l) \\ \hat{X}_{k-1}^S(2i, 2j+1, l) & = (X_{k-1}^S(i, j, l) + X_{k-1}^S(i, j+1, l))/2 \\ \hat{X}_{k-1}^S(2i+1, 2j, l) & = (X_{k-1}^S(i, j, l) + X_{k-1}^S(i+1, j, l))/2 \\ \hat{X}_{k-1}^S(2i+1, 2j+1, l) & = (X_{k-1}^S(i, j, l) + X_{k-1}^S(i+1, j+1, l))/2 \end{cases} \tag{4.21}$$

where $i = 0, 1, ...M/2^{p-k+1} - 1$, $j = 0, 1, ...N/2^{p-k+1} - 1$,
and $l = 0, 1, ...L - 1$.

(b) The interpolated level, $\hat{X}^S_{k-1}$, is subtracted from the corresponding nodes at level k,

$$\begin{cases} E^S_k(2i, 2j, l) & = X^S_k(2i, 2j, l) - \hat{X}^S_{k-1}(2i, 2j, l) \\ E^S_k(2i, 2j+1, l) & = X^S_k(2i, 2j+1, l) - \hat{X}^S_{k-1}(2i, 2j+1, l) \\ E^S_k(2i+1, 2j, l) & = X^S_k(2i+1, 2j, l) - \hat{X}^S_{k-1}(2i+1, 2j, l) \\ E^S_k(2i+1, 2j+1, l) & = X^S_k(2i+1, 2j+1, l) - \hat{X}^S_{k-1}(2i+1, 2j+1, l) \end{cases} \tag{4.22}$$

where $i = 0, 1, ...M/2^{p-k+1} - 1$, $j = 0, 1, ...N/2^{p-k+1} - 1$,
and $l = 0, 1, ...L - 1$.

Since, from Eqn. 4.3 and Eqn. 4.21,

$$E^S_k(2i, 2j, l) = 0, \tag{4.23}$$

and hence, $E_k(2i, 2j, l)$ can be discarded from the spatial prediction difference pyramid.

2. *For the top level $k = 1$:*

$$E^S_k(i, j, l) = X^S_k(i, j, l). \tag{4.24}$$

Fig. 4.4 illustrates a simple example of spatial prediction pyramid formation where the even number nodes of the difference pyramids are discarded. As each level discards 1/4th of its nodes, the total number of nodes is now equal to the number of pixels of the original video sequence. The spatial prediction difference pyramid consists of the spatial prediction differences and therefore has less spatial correlation.

The spatial sample pyramid can be reconstructed from the spatial prediction difference pyramid starting from the top level, $k = 1$, as follows:

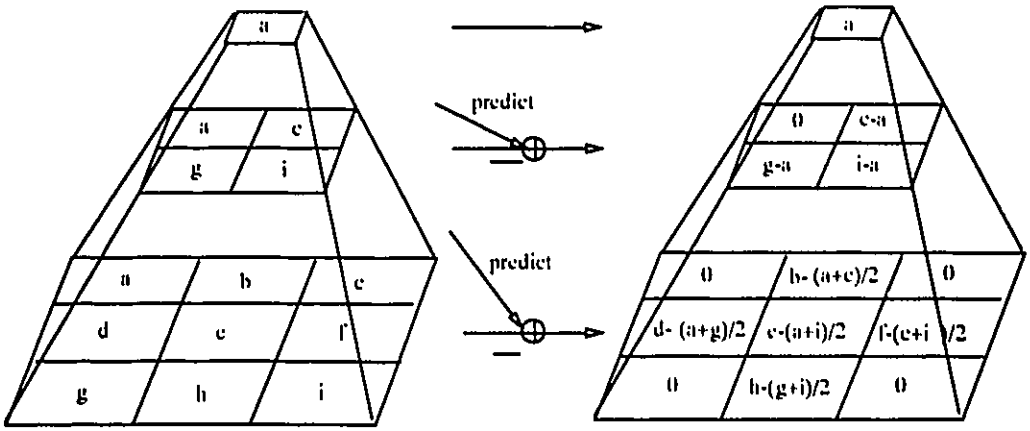

Figure 4.4: (a) Spatial sample pyramid (b) Spatial prediction difference pyramid

1. *For the top level* $k = 1$*:*

$$X_k^S(i,j,l) = E_k^S(i,j,l). \tag{4.25}$$

2. *For level* $k = 2, 3, ...p$*:* Level $k-1$ are spatially interpolated using linear prediction as given in Eqn. 4.21 and added to the corresponding difference nodes at level k,

$$\begin{cases} X_k^S(2i,2j,l) & = \hat{X}_{k-1}^S(2i,2j,l) + 0 \\ X_k^S(2i,2j+1,l) & = \hat{X}_{k-1}^S(2i,2j+1,l) + E_k^S(2i,2j+1,l) \\ X_k^S(2i+1,2j,l) & = \hat{X}_{k-1}^S(2i+1,2j,l) + E_k^S(2i+1,2j,l) \\ X_k^S(2i+1,2j+1,l) & = \hat{X}_{k-1}^S(2i+1,2j+1,l) + E_k^S(2i+1,2j+1,l) \end{cases} \tag{4.26}$$

where $i = 0, 1, ...M/2^{p-k+1} - 1$, $j = 0, 1, ...N/2^{p-k+1} - 1$, and $l = 0, 1, ...L-1$.

The base level, X_p^S, is the original video sequence, X, i.e., $X_p^S = X$.

4.4 Adaptive Temporal/Spatial Pyramid

It has been shown that the temporal and spatial difference pyramids are more suitable data structures for coding purposes compared to the temporal and spatial sample pyramids because the difference pyramids have less data to be coded and have less correlation in both the temporal and spatial domains. The interesting question is which of the two difference pyramids should be used for a specific video sequence. We recall from chapter 2 that there are two basic correlations in a video sequence: temporal and spatial. The temporal and spatial difference pyramids address the temporal and spatial correlations, respectively. We propose an adaptive 3D pyramid in which a pyramid level can be contracted in either the temporal or spatial dimension, depending on the correlation present in a video sequence. If the video signal demonstrates a higher degree of temporal correlation, the temporal difference pyramid is used; otherwise, we employ the spatial difference pyramid.

The procedure adopted for adaptively selecting temporal or spatial decimation is as follows:

1. Contract the signal in temporal and spatial dimesions independently.
2. Obtain the corresponding difference (or prediction difference) levels.
3. Compare the temporal and spatial differences.
4. If the temporal difference is lower than the spatial difference, then the next level is derived by temporal decimation, otherwise by spatial decimation.
5. Repeat the steps 1 to 4 (for the next level) until the desired number of levels are obtained.

An example of a 3D adaptive pyramid formed by time, time, space (T, T, S) decimation for 8 frames of the video signal is shown in Fig. 4.5. The difference pyramid

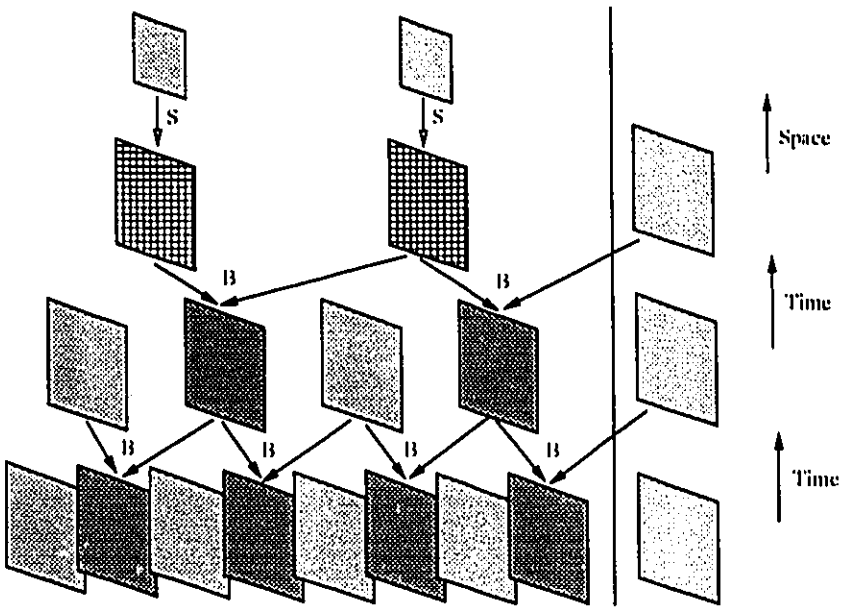

Figure 4.5: A 3D adaptive pyramid formed by time, time and space decimations [S: Spatial prediction B: Bidirectional temporal prediction]

is simply the difference between two adjacent levels as defined in section 4.2 and 4.3. Here, the arrows show the use of temporal bidirectional prediction and spatial non-causal linear prediction as defined in Eqns. 4.15 and 4.21. In other words, in case of a temporal decimation, bidirectional motion compensated prediction (B) is employed whereas in case of a spatial decimation, non-causal prediction (S) is employed to obtain the difference level in the pyramid.

Data	*Details*		
Image Seq.	Size	Entropy	Fig No.
1. Miss America	288 × 352	5.95	5.12
2. Salesman	288 × 352	6.85	5.20
3. Albert	208 × 256	5.35	5.19

Table 4.1: Simulation test data

4.5 Performance Comparison of Different Pyramids

The temporal difference, spatial difference, adaptive temporal/spatial difference and non-adaptive difference pyramids are simulated and the performance is studied for video compression. Test data described in Table 4.1 are used for the simulations. The performance is measured in terms of first order entropy as defined in Eqn. 2.7. From the entropy we can measure the compactness of the pyramid and its usefulness for video compression. The MSE of the difference level is used to select between the temporal and spatial decimations. We have performed three sets of experiments. We note that for the first two sets of experiments, space, time alternate and time, space alternate decimations are used as non-adaptive pyramids. The first set of experiments was performed using the difference pyramids for the Salesman and the Miss America sequences. The results are reported in Table 4.2.

Some observations on the simulation results are in order. The time, space decimations in different configurations have different entropies which suggests that the adaptive temporal/spatial decimation should have the lowest entropy. As shown in Table 4.2, the adaptive pyramid has the lowest entropy which justifies the use of

Contraction types	*Entropy*	
Difference Pyramids	Salesman	Miss America
1. Spatial	5.51	4.35
2. Temporal	4.15	4.16
3. Time, Space Alt.	4.77	4.19
4. Space, Time Alt.	5.24	4.27
5. Adaptive	4.15	4.16

Table 4.2: Entropy of various difference pyramids

Contraction types	*Entropy*	
Prediction Difference Pyramids	Salesman	Miss America
1. Spatial	5.13	4.00
2. Temporal	4.04	3.75
3. Time, Space Alt.	4.50	3.78
4. Space, Time Alt.	4.90	3.94
5. Adaptive	4.04	3.75

Table 4.3: Entropy of various prediction difference pyramids

Types of Control	*Scene Change1*		*Scene Change2*	
Pyramids	Selected Contr.	Entropy	Selected Contr.	Entropy
1. Adaptive	T, T, S	3.88	T, T, S	3.92
2. Non-adaptive	T, T, T	4.60	T, T, T	4.50
3. Non-adaptive	S, S, S	4.00	S, S, S	4.63

Table 4.4: Entropy of adaptive/non-adaptive pyramids for scene changes

adaptivity in the pyramid.

The second set of experiments was performed using the prediction difference pyramids. The results are shown in Table 4.3. The temporal prediction reduces the entropy of the temporal pyramid by 0.4 bits/pixel for the Miss America sequence but for the Salesman sequence it does not help very much. We note that there is an overhead for the transmission of the motion information (of about 0.10 bits/pixel) in the case of the temporal prediction. The use of spatial prediction reduces the spatial pyramid entropy by 0.35-0.40 bits/pixel for the Miss America and the Salesman sequences. These results justify the use of predictions in pyramids. It can be seen that the 3D adaptive prediction difference pyramid gives the lowest entropy.

It is interesting to confirm the results for the sequences with scene changes. In the third experiment, four frames of the Albert sequence and four frames of the Miss America sequence are used for the first scene change and that of the Miss America sequence and the Salesman sequence are used for the second scene change. The results are presented in Table 4.4 which show that for scene changes, the adaptive pyramid has lower entropy compared to the non-adaptive pyramid. Thus, the pyramid performs very well even during the scene changes.

Finally, the histogram distribution of the pyramids with and without the use of the non-causal prediction techniques are presented. Fig. 4.6 shows the histogram of the temporal pyramids with and without temporal motion compensated prediction. As seen, temporal prediction difference pyramid has a larger number of pixels concentrated around zero. Fig. 4.7 shows the histogram of the spatial pyramids with and without spatial non-causal linear prediction. The spatial prediction difference pyramid also has a larger number of pixels containing gray values in the range [-10, 10]. Thus, pyramids employing prediction techniques are more compact and have a lower first order entropy compared to the pyramids without predictions.

4.6 Conclusions

In this chapter, algorithms to form the temporal/spatial pyramids for video compression are proposed which include the temporal sample and spatial sample pyramids. Difference pyramids are used for coding purposes. Two methods to obtain temporal difference and spatial difference pyramids are discussed. One of these methods employs non-causal prediction techniques in both the temporal and spatial domains. Performance of the various pyramids are compared for video compression in terms of the first order entropy. Pyramids using predictions have a lower first order entropy and are compact. An adaptive pyramid data structure is proposed which selects either the temporal or the spatial decimation based on the prediction difference. It has been shown that the adaptive pyramid achieves the lowest first order entropy and performs well even during the scene changes.

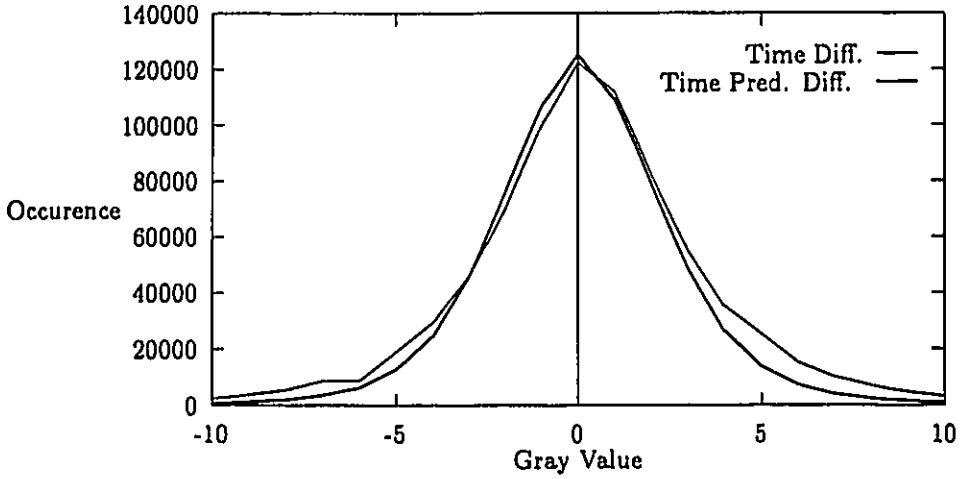

Figure 4.6: Histogram of the temporal difference pyramid with and without prediction

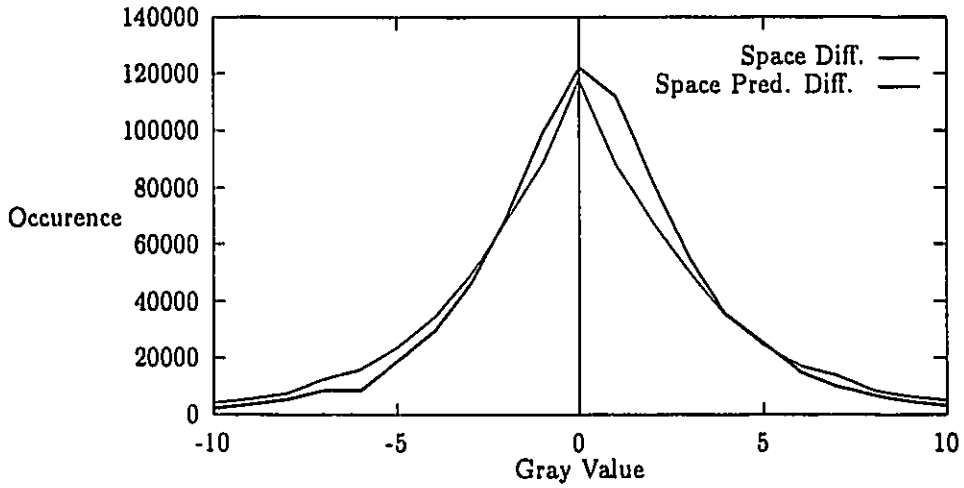

Figure 4.7: Histogram of the spatial difference pyramid with and without prediction

Chapter 5

Video Coder using a 3D Adaptive Pyramid

In this chapter, a video coder based on the 3D adaptive temporal/spatial pyramid is presented. We recall from chapter 2 that the hybrid coder employed in the MPEG standard algorithm exploits the temporal correlation using inter-frame DPCM and the spatial correlation using intra-frame DCT. The proposed video coder not only exploits the temporal and spatial correlations as in the MPEG coder but also exploits the spatio-temporal correlation using the 3D adaptive pyramid data structure. In the 3D adaptive pyramid, a level is contracted in either the temporal or spatial domain based on the corresponding prediction differences. The 3D adaptive pyramid is encoded using a lossy intra-frame coding technique (universal vector quantization [UVQ]) to achieve further compression. Here, the accumulation of the coding errors is avoided by delivering these errors to the lower levels of the pyramid. Error delivery provides the possibility of lossless coding which is useful in many applications. A hierarchical buffer control scheme is employed in combination with a bit allocation method which results in a constant bit rate output video signal. Simulation results of the proposed pyramidal video coder are presented for video conferencing and HDTV applications, and compared with the baseline MPEG standard coder. Finally, the significant features of the pyramidal coder are summarized.

5.1 Video Coder using a 3D Adaptive Pyramid

The proposed video coder based on the 3D adaptive temporal/spatial prediction difference pyramid is shown in Fig. 5.1. We have designed and implemented this video coder. The implementation details are presented in section 5.4. We recall from chapter 2 that a video signal has two basic correlations; temporal and spatial. We also recall that the hybrid coder used in the MPEG standard employs temporal and spatial coding techniques to remove the corresponding correlations. The proposed video coder exploits the temporal correlation by motion compensation, the spatio-temporal

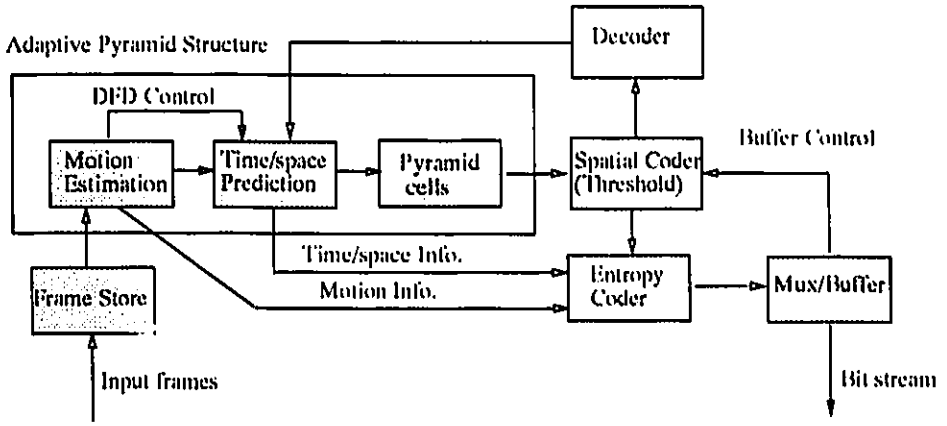

Figure 5.1: Video encoder using a 3D adaptive temporal/spatial pyramid

correlation using the 3D adaptive pyramid and the spatial correlation using an intra-frame coder. The 3D adaptive pyramid switches between the temporal and spatial pyramid based on the temporal and spatial prediction differences. The pyramidal coder has three main units: temporal, spatio-temporal and spatial. The temporal unit uses a motion compensation prediction technique similar to the MPEG coder to exploit the temporal correlation present in the video signal. The spatio-temporal unit has an overlap between the temporal and spatial correlations and consists of two sub-units; temporal/spatial predictor and pyramid cells. The spatial unit is essentially an intra-frame coder, a block classifier and an entropy coder as in the MPEG coder. The functional details of these units follows.

Temporal Unit

The temporal unit exploits the temporal correlation using the motion compensation technique. Here, the input video sequence from the frame store is divided into groups of pictures (GOP) which is stored at the bottom level of the pyramid cells. Motion estimation is performed on the GOP using the full search block matching algorithm.

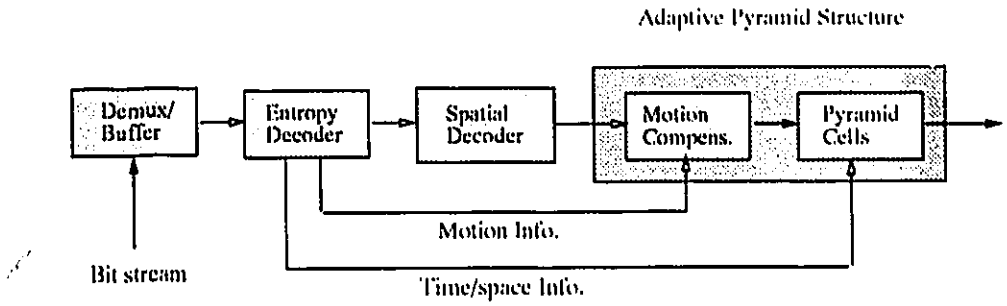

Figure 5.2: Video decoder using a 3D adaptive temporal/spatial pyramid

Whenever possible bidirectional motion estimation is employed in the pyramid and a bit-plane is transmitted to the receiver for forward/backward motion compensation. We note that the bidirectional motion estimation deals with problems such as covered/uncovered areas, background noise, illuminance change, etc. At the top level of the pyramid, unidirectional motion estimation is employed.

Spatio-temporal Unit

In this unit, each GOP is reorganized into the 3D adaptive prediction difference pyramid format as discussed in section 4.4. The predictor uses the motion estimation information to decide between temporal and spatial predictions. We note that in the temporal domain, motion compensated prediction and in the spatial domain, non-causal linear prediction are employed. The 3D difference pyramid is stored in the pyramid cells.

Spatial Unit

The remaining correlations present in the difference pyramid is exploited by encoding the pyramid using a lossy intra-frame coder in the spatial unit which achieves further data compression. The spatial coder consists of an intra-frame coder, block classi-

fication sub-unit and an entropy coder. Coding techniques such as universal vector quantization (UVQ), variable length DCT (VLDCT), etc. can be used as an intra-frame coder. In this thesis, the UVQ is employed to encode the pyramid. We recall from section 2.3.3 that UVQ is an efficient technique at low bit rate video coding. The coding errors introduced by the lossy coder are delivered to the lower levels of the pyramid to avoid error accumulation as discussed in section 5.2. The vectors in the UVQ are classified into significant/non-significant vectors before encoding. Features such as energy, variance, mean, etc. of the vectors can be used to find non-significant vectors [38]. In this thesis, the energy criterion is used as it directly corresponds to MSE of the reconstructed signal. The vectors below some threshold value of the energy are discarded. This helps to reduce the noise in the uniform regions in the case of wrong predictions. The percentage of significant vectors for different threshold values are plotted in Fig. 5.8. It can be seen that for a threshold of 10, more than 50 % of the vectors are non-significant. The information about significant/non-significant blocks is transmitted to the receiver in the form of a bit-plane. We note that the target bit rate for a particular level can be achieved by varying the threshold of the significant/non-significant block classification.

The entropy coder assigns the binary codes for the quantized information. The final level of the pyramid is coded using Huffman coding technique when lossless compression is required. In addition, motion vectors and labels of the UVQ can be coded using Huffman coding in order to exploit their non-uniform distributions. The multiplexer synchronizes different information and an output bit stream is generated. The buffer is constantly monitored using a buffer control signal and upon buffer fullness, the final level of the pyramid is coarsely quantized. This results in a more uniform picture quality.

Adaptivity Control Signal

The proposed 3D adaptive pyramid switches between the temporal and spatial pyramids based on the temporal and spatial prediction differences. The MSE of the temporal and spatial prediction differences are compared to adapt the coder as described in section 4.4. This requires the computation and storage of the spatial prediction differences. The method can be simplified if the predictor is switched to the spatial pyramid when motion estimation fails. This is made possible by using the displaced frame difference (DFD) which is calculated during the motion estimation. Since the DFD directly corresponds to the temporal prediction difference energy, its value can be approximated by some experiments on temporal/spatial prediction, above which a control signal informs the predictor that the motion estimation is unsuccessful. The temporal/spatial predictor then switches to the spatial prediction and checks the spatial prediction difference energy.

A control signal is generated using the displaced frame difference called, DFDC, which is defined as follows:

$$\begin{aligned} \text{DFDC} &= F\frac{1}{K}\sum_{k=0}^{K-1} MSE_k \\ &= F\frac{1}{[K\times M/K\times N/K]}\sum_{k=0}^{K-1}\sum_{i=0}^{M/K-1}\sum_{j=0}^{N/K-1}(X_{ij}-X'_{ij})^2 \\ &= F\frac{K}{M\times N}\sum_{k=0}^{K-1}\sum_{i=0}^{M/K-1}\sum_{j=0}^{N/K-1}(X_{ij}-X'_{ij})^2 \end{aligned} \qquad (5.1)$$

where X_{ij} is the original pixel, X'_{ij} is the predicted pixel, MSE is the same as defined in Eqn. 2.22, K is the total number of blocks and F = 1/100 is a normalization factor. The MSE is the minimum value found for the matched block during the motion estimation operation and the minimum block MSE's over all the blocks in a frame are averaged. In this thesis, DFDC > 1 is used to switch to the spatial contraction. A normalization factor F = 1/100 makes DFDC = 1, for a MSE of 100. For the Albert sequence, DFDC = 1 corresponds to a NMSE of 1 %. The simulation results of the adaptive pyramid entropy suggest that the DFDC signal yields comparable performance to the MSE comparison method. Fig. 5.6 shows the

DFDC for the unidirectional and the bidirectional full search motion estimations. As shown, the bidirectional motion estimation results in almost constant DFDC, whereas the unidirectional motion estimation DFDC varies depending on the motion in the sequence. The MSE of a spatial pyramid normalized by 100 is shown in Fig. 5.7 for comparison.

DPCM Coding of the Motion Vectors

We recall from section 2.4.1 that in the block matching algorithms (BMA), motion estimation is performed on a block basis. Each frame is divided into the microblocks of size 8×8 and motion vectors are obtained for each microblock. These motion vectors (MV) are lossless DPCM coded i.e. a new MV is obtained from the microblock vector by subtracting the vector of the preceding microblock. The vector of the preceding microblock is regarded as zero in the case of the first microblock of the frame. A variable length code (VLC) table for the motion vectors for a maximum displacement of 6, is designed as shown in Table 5.2, which is similar to the one given in [18]. It can be seen that each VLC represents a pair of difference values. Here, only one vector from the pair if added to the previous MV will yield a new MV falling within the range [-6, 6]. For example, if the code 0011 which corresponds to (-2, 20) is added to the previous MV (say, 5), a new pair of (3, 25) is obtained in which only MV, 3 is within the permitted range.

The algorithmic steps of the pyramidal video coder can be summarized as follows.

begin pyramid

Intitialize $k = 1$.

Let p be the number of pyramid levels.

Divide the input frames into group of frames (GOP).

Store the GOP in the pyramid cells, at level X_k.

Do

- *Perform motion estimation on the pyramid level X_k.*
- *Find DFDC along with motion estimation.*
- *If (DFDC > 1) contract the level X_k in the spatial domain*
  else contract X_k in the temporal domain.
- *Store the contracted level in the pyramid cells at X_{k+1}.*

while (k ≤ p)

end pyramid

begin encoding

Let k = p, i.e. the number of pyramid levels.

Do

- *Encode the pyramid level, X_k, using the pre-allocated bits.*
- *Store the decoded level, $\hat{X}_k$, of the pyramid in the decoder.*
- *Predict the next pyramid level X_{k-1} using the decoded level.*
- *Deliver the encoding errors to the prediction difference level, E_{k-1}.*
- *Replace the level in the pyramid cells with the prediction difference level, E_{k-1}.*

while (k > 0)

Based on the available bit rate, the final level is coarsely or finely quantized.

end encoding

Comparison of Pyramid and MPEG-1 Structures

The proposed pyramidal video coder exploits the spatial-temporal correlation present in the video signal more effectively than the MPEG-1 coder as shown in section 5.4. In addition, the video coder has a number of structural similarities with the MPEG-1 video compression standard. Both methods work on the GOP approach. The 4-level temporal pyramid is an extension of the 2-level MPEG frame organization. We note that some of the recommendations of the MPEG committee have been incorporated

into the coder. Bidirectional motion estimation as well as the concepts of I, P, and B frames with proper bit allocations are employed. The pyramid structure makes the buffer control very simple compared to the MPEG coding structure. The bit-stream is made compatible to the MPEG bit stream with the order of I, P, and B frames. The pyramid structure can be mapped onto the MPEG hardware with minor modifications. We have used an alternative coding technique based on the observation that VLDCT does not perform very well for the error images [33]. The UVQ technique has the advantages that the decoder is very simple and requires only table look-up operations. Finally, the proposed coder exhibits most of the required features for the video coder as discussed in [20] for the MPEG standard.

5.2 Lossless Coding

In the proposed video coder, the 3D adaptive pyramid is encoded using a lossy intra-frame coding technique to achieve further compression. The lossy coding at the upper levels of the pyramid results in the accumulation of the errors which degrades the coding performance. These errors can be delivered to the lower levels of the pyramid. The error delivery property ensures that the correct values are reconstructed. If the final level of the pyramid is transmitted without any coding errors, the signal can be losslessly reconstructed. In the proposed 3D adaptive pyramid, the information about temporal/spatial contraction of the pyramid is required for proper error delivery. In other words, if the contraction is spatial, then the spatial error delivery is employed, otherwise the temporal error delivery is employed.

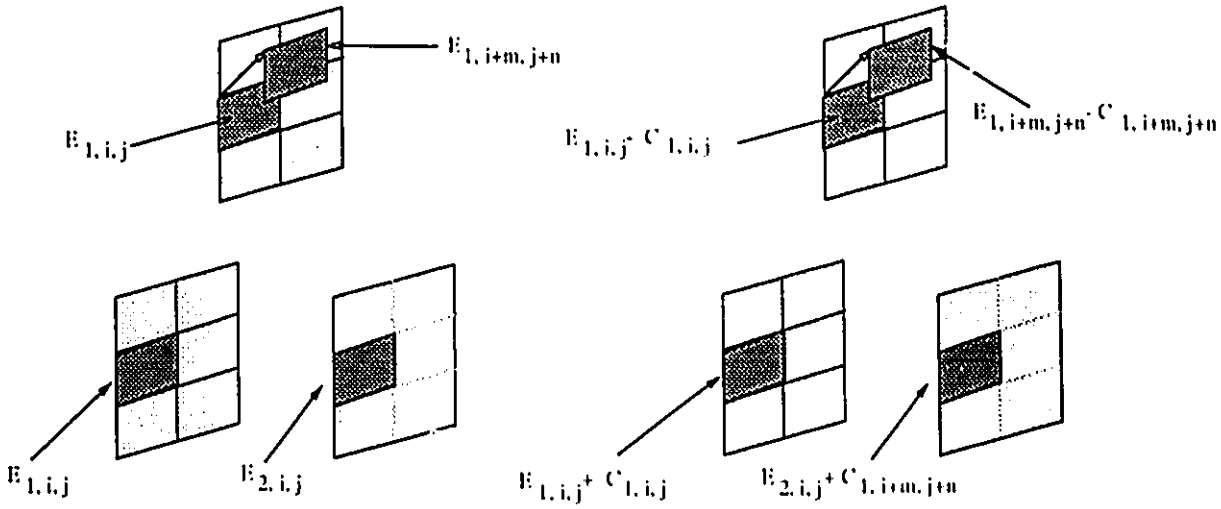

Figure 5.3: Temporal error delivery in the temporal difference pyramid

5.2.1 Temporal Error Delivery

The lossy coding technique employed to code the temporal contraction levels introduces coding errors. These errors can be fed-forward to the lower levels of the pyramid to be reprocessed. We note that the temporal error delivery also uses the motion compensation technique. The coding errors introduced at each level, $k-1$, of the temporal pyramid are calculated and added to the respective microblocks at the next level. For the temporal prediction difference pyramid (Eqn. 4.15), the coding errors introduced at level $k-1$ are given by,

$$\begin{cases} C^T_{k-1}(i,j,2l) & = X^T_{k-1}(i,j,l) - \hat{X}^T_{k-1}(i,j,l) \\ C^T_{k-1}(i,j,2l+1) & = X^T_{k-1}(i+m,j+n,l+o) - \hat{X}^T_{k-1}(i+m,j+n,l+o) \end{cases} \tag{5.2}$$

where $i = 0,1,2,...M-1$, $j = 0,1,2,...N-1$, and $l = 0,1,...L/2^{p-k+1}-1$. Note that (m,n,o) are the motion vectors obtained by bi-directional motion estimation.
The encoding errors, C^T_{k-1}, are then added to the corresponding microblocks at level

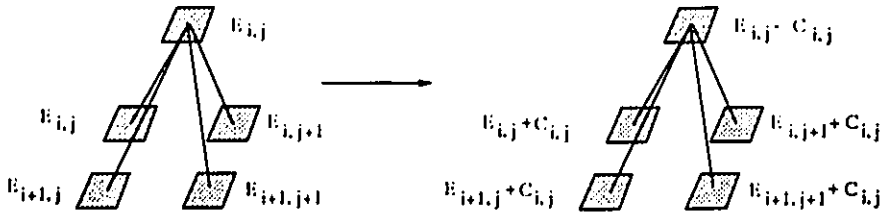

Figure 5.4: Spatial error delivery in the spatial difference pyramid

k, as given by,

$$\begin{cases} \hat{E}_k^T(i,j,2l) & = E_k^T(i,j,2l) + C_{k-1}^T(i,j,2l) \\ \hat{E}_k^T(i,j,2l+1) & = E_k^T(i,j,2l+1) + C_{k-1}^T(i,j,2l+1) \end{cases} \tag{5.3}$$

where $i = 0,1,2,...M-1$, $j = 0,1,2,...N-1$, and $l = 0,1,...L/2^{p-k+1}-1$.
We note that the original signal can be recovered from the temporal prediction difference pyramid if the final modified error level, $\hat{E}_k^T$, is losslessly coded using techniques such as the Huffman coding, arithmetic coding, etc.

5.2.2 Spatial Error Delivery

In a spatial pyramid, the lossy coding technique employed to code the upper levels introduces errors which accumulates as the lower levels are coded. The coding errors introduced at each level, $k-1$, of the spatial pyramid are delivered to the next level, k, and added to the respective sibling nodes. For the spatial prediction difference pyramid (Eqn. 4.21), the coding errors introduced at level $k-1$ are given by,

$$\begin{cases} C_{k-1}^S(2i,2j,l) & = X_{k-1}^S(i,j,l) - \hat{X}_{k-1}^S(i,j,l) \\ C_{k-1}^S(2i,2j+1,l) & = \frac{X_{k-1}^S(i,j,l)+X_{k-1}^S(i,j+1,l)}{2} - \frac{\hat{X}_{k-1}^S(i,j,l)+\hat{X}_{k-1}^S(i,j+1,l)}{2} \\ C_{k-1}^S(2i+1,2j,l) & = \frac{X_{k-1}^S(i,j,l)+X_{k-1}^S(i+1,j,l)}{2} - \frac{\hat{X}_{k-1}^S(i,j,l)+\hat{X}_{k-1}^S(i+1,j,l)}{2} \\ C_{k-1}^S(2i+1,2j+1,l) & = \frac{X_{k-1}^S(i,j,l)+X_{k-1}^S(i+1,j+1,l)}{2} - \frac{\hat{X}_{k-1}^S(i,j,l)+\hat{X}_{k-1}^S(i+1,j+1,l)}{2} \end{cases} \tag{5.4}$$

where $i = 0, 1, 2, ...M/2^{p-k+1} - 1$, $j = 0, 1, 2, ...N/2^{p-k+1} - 1$, and $l = 0, 1, ...L - 1$. The encoding errors, C_{k-1}, are then added to the corresponding nodes at level, k, as given by,

$$\begin{cases} \hat{E}_k^S(2i, 2j, l) & = E_k^S(2i, 2j, l) + C_{k-1}^S(2i, 2j, l) \\ \hat{E}_k^S(2i, 2j+1, l) & = E_k^S(2i, 2j+1, l) + C_{k-1}^S(2i, 2j+1, l) \\ \hat{E}_k^S(2i+1, 2j, l) & = E_k^S(2i+1, 2j, l) + C_{k-1}^S(2i+1, 2j, l) \\ \hat{E}_k^S(2i+1, 2j+1, l) & = E_k^S(2i+1, 2j+1, l) + C_{k-1}^S(2i+1, 2j+1, l) \end{cases} \tag{5.5}$$

where $i = 0, 1, ...M/2^{p-k+1} - 1$, $j = 0, 1, ...N/2^{p-k+1} - 1$, and $l = 0, 1, ...L - 1$. Similar to the temporal difference pyramid, the original signal can be recovered from the spatial difference pyramid, if the final error level, $\hat{E}_k^S$, is losslessly coded using techniques such as the Huffman coding, arithmetic coding, etc.

5.3 Hierarchical Buffer Control Scheme

We recall from section 2.6 that buffer control is very important in low bit rate video coding applications. In the proposed pyramidal video coder, the bit rate varies depending on the correlation present in the video signal. For example, high detail frames produce large number of significant blocks, thus increasing the bit rate. In addition, the variable length Huffman codes result in rapid bit rate variations and hence, a larger buffer size is required. In this section, we describe a hierarchical buffer control scheme which results in a constant bit rate video. In this scheme, fixed bits are assigned to the different levels of the pyramid and an iterative technique is used to achieve the target bit rate. In the iterative technique, the output bit rate is compared with the target bit rate. The threshold of the block classification unit is varied until the target bit rate is achieved. If the target bit rate is high, then the threshold of the block classification is decreased resulting in less number of nonsignificant blocks.

The comparison and threshold variation process continues until the resulting bit rate is close to the target bit rate. Since the video signals usually are slowly changing, the number of iterations required for determining the threshold is very small.

The procedure adopted for determining the threshold for a frame, $k+1$, in a pyramid level, p, can be summarized in algorithmic form as follows:

1. Set the iteration index, $i = 0$, and the bit rate difference, $\Delta_{i-1} = 0$. Initialize the threshold δ_i by the value used for the previous frame at the same pyramid level p or the previous level $p-1$.

2. Apply the threshold, δ_i, to the current frame, resulting in an output bit rate, B_i.

3. Calculate the bit difference between the output bit rate, B_i, and the target bit rate, B_{target},

$$\Delta_i = B_i - B_{target} \tag{5.6}$$

   If the sign of the bit rate difference, Δ_i, is different than the previous value, Δ_{i-1}, go to step 5; otherwise continue.

4. If

$$\Delta_i > 0 \tag{5.7}$$

   increase δ_i by one unit, otherwise, decrease δ_i by one unit.
   Let $i = i + 1$. Go to Step 2.

5. Let accumulated bit rate difference up to the current frame, k, at current pyramid level p be,

$$\Delta B_{total}(k) = \sum_{l=0}^{k} \Delta B(l) \tag{5.8}$$

   where $\Delta B(l)$ is the difference between the actual output bit rate for frame l, $B_{output}(l)$ and the target bit rate, B_{target}, i. e.

$$\Delta B(l) = B_{output}(l) - B_{target} \tag{5.9}$$

If

$$|\Delta B_{total}(k) + \Delta_i| < |\Delta B_{total}(k) + \Delta_{i-1}|, \tag{5.10}$$

select δ_i as the final threshold, otherwise, δ_{i-1}.

The output bit rate, bit rate difference and total accumulated difference in bit rate up to frame k+1 at pyramid level, p, is updated using the threshold value obtained for the frame k+1.

5.3.1 Bit Allocation and Target Bit Rate

Output bit rate variations in the proposed video coder necessitates the use of large size buffer. Buffer control can be made simple by proper bit allocation. In addition, bit allocation helps to generate an output video sequence with constant quality. We recall from chapter 4 that the 3D adaptive pyramid consists of I, P and B frames. In the I frames, only intra-frame correlation is exploited whereas in the P and B frames, the inter-frame as well as intra-frame correlations are exploited. Generally, the I frames require 2 to 3 times more bits than the P frames which in turn require 2 to 3 times more bits than the B frames. Based on this criteria, we can allocate the bits appropriately to the different levels, after the pyramid has been formed. If $B_{available}$ is the available bit rate per GOP, then

$$Bit\ rate\ for\ the\ I\ frames\ (B_I) = \frac{\alpha_1\alpha_2}{\alpha_1\alpha_2 N_1 + \alpha_2 N_2 + N_3} B_{available} \tag{5.11}$$

$$Bit\ rate\ for\ the\ P\ frames\ (B_P) = \frac{\alpha_2}{\alpha_1\alpha_2 N_1 + \alpha_2 N_2 + N_3} B_{available} \tag{5.12}$$

$$Bit\ rate\ for\ the\ B\ frames\ (B_B) = \frac{1}{\alpha_1\alpha_2 N_1 + \alpha_2 N_2 + N_3} B_{available} \tag{5.13}$$

Here, the number of I, P and B frames are N_1, N_2 and N_3, respectively. α_1 and α_2 are respectively, the ratios of the bits assigned to the I, P frames and P, B frames. Note that the sum of bit rates is $B_{available}$.

For a particular level, the bits are allocated depending on the number of the I, P and B frames at that level. For example, the bit assignment for the pyramid with T, T, S contractions is calculated as follows:

1. Let us assume $\alpha_1 = \alpha_2 = 2$.

2. From Fig. 4.5, $N_1 = 2, N_2 = 2, N_3 = 4$.

3. From Eqns. 5.11, 5.12 and 5.13, the following bit rates can be obtained,

$$B_I = \frac{4}{16} B_{available} \tag{5.14}$$

$$B_P = \frac{2}{16} B_{available} \tag{5.15}$$

$$B_B = \frac{1}{16} B_{available} \tag{5.16}$$

4. The bit assignment to the different pyramid levels is as follows:

$$Level\ 4 + Level\ 3 = 2 \times B_I = \frac{8}{16} B_{available} \tag{5.17}$$

$$Level\ 2 = B_P + B_B = \frac{3}{16} B_{available} \tag{5.18}$$

$$Level\ 1 = 3 \times B_B + B_P = \frac{5}{16} B_{available} \tag{5.19}$$

We note that in the spatial pyramid, equal bits are assigned to the different frames in the pyramid. Here, unused bits in one frame are used in the next frame of the same level, and unused bits of one level are used in the next level. Upon buffer fullness, the final level of the pyramid can be skipped and interpolated at the receiver. This bit allocation mechanism has a simple buffer control and ensures uniform picture quality.

5.4 Simulation Results

The proposed pyramidal video coder has been simulated and the performance studied for the CCITT monochrome test sequences as shown in Table 4.1. The Miss America sequence has low spatial details and moderate motion whereas the Salesman sequence has higher spatial details and low motion. The Calendar sequence has very high spatial details and moderate motion. We now present the simulation results and details. As discussed in section 5.1, the universal vector quantization (UVQ) technique is employed as a spatial coder. We have generated separate universal codebooks for the different levels of the pyramid as well as for the temporal and spatial difference levels using the LBG algorithm. Since the top level of the pyramid has original pixel values, a universal codebook of size 512 and dimension 16 was generated from ten different still images. For the difference levels, the video coder was simulated on the Salesman and Albert sequences and a training sequence of 16 error frames was formed. Using this training sequence, various codebooks of size 256 and dimension 16 were generated. Preliminary simulations using the MPEG-1 coder have been performed and used as a baseline for comparison. Details of the coder have been presented in sections 2.4.2 and 2.5.1. We note that no attempt has been made to generate particular bit stream [21] as the purpose of this simulation is to compare the performance of the different structures. Six sets of experiments have been performed using the proposed coder. The first five sets of experiments were performed using the video conferencing test sequences given in Table 4.1 whereas the last set of experiments was performed using the CCIR resolution test sequences for the HDTV.

In the first set of experiments, the performance of the 3D adaptive pyramid is compared with that of the non-adaptive pyramids. With a lossy coder (UVQ), different configurations of the temporal/spatial contractions give different PSNR values as shown in Table 5.1 for the Miss America sequence. The adaptive pyramid has up to 3 db higher PSNR than the non-adaptive pyramid. The results reported are for

Contraction types	*PSNR*
Prediction difference pyramid	Miss America
1. Spatial	34.0
2. Temporal	36.5
3. Time, Space Alt.	34.9
4. Space, Time Alt.	33.5
5. Adaptive	36.5

Table 5.1: PSNR values of different prediction difference pyramids for the Miss America sequence

the pyramids using prediction. We have found that the use of prediction in pyramids improves the PSNR values in the range of 0.2-0.6 db for the first 8 frames of the Miss America sequence.

In the second set of experiments, simulation results are obtained for 96 frames of the Miss America sequence and shown in Fig. 5.9. The average bits/pixel is 0.5 corresponding to a bit rate of 1.5 Mb/s. We note that this bit rate includes the bits for the UVQ index, significant/non-significant block information and the motion information. The average PSNR value obtained is 36.6 db. The original frame no. 4 and the corresponding reconstructed frame of the Miss America sequence are shown in Figs. 5.12 and 5.13, respectively. The error frame enhanced by histogram stretching method (i.e. shifted by 128, multiplied by 255 and divided by (maximum gray value - minimum gray value)) is shown in Fig. 5.14 for subjective evaluation. It can be seen that overall subjective quality of the coded frame is excellent although the high details are not reproduced well especially near the lips and the eyes. The edges can be enhanced by using techniques such as classified vector quantization [67]. For the sake of comparison, the corresponding reconstructed and enhanced error frames

of the Miss America sequence using the spatial pyramid are shown in Fig. 5.15 and 5.16, respectively. Here, the edges are more fuzzy and there is a loss in the gross structure of the image. The drop in the PSNR value near frames 75-85 is the result of the unidirectional motion estimation employed at the top level of the pyramid as well as fast motion in the sequence. We note that this PSNR value (Fig. 5.9) is still higher than the corresponding spatial pyramid by 0.5 db. Thus, this adaptive temporal/spatial pyramid exploits either the temporal or the spatial correlation, whichever is higher.

In the third set of experiments, the performance of the proposed coder is confirmed for the sequences with scene changes and compared with that of the MPEG coder. The test sequence with the scene change consists of 12 frames of the Miss America sequence and 12 frames of the Salesman sequence. We note that the 24 frames are arranged in three different pyramids as each pyramid has 8 frames (GOP). The second pyramid includes the scene change. The adaptive pyramid has T, T, S contraction configuration. The PSNR values of the frames are shown in Fig. 5.10. It can be seen that there is a smooth transition from one scene to the next scene and the PSNR value of the first frame of the Salesman sequence is not low compared to the average PSNR of the sequence. Thus, the proposed pyramid adapts to the scene changes. The original and the reconstructed frames of the Salesman sequence are shown in Fig. 5.20 and 5.21, respectively which is the second frame after the scene change from the Miss America sequence. The corresponding enhanced error frame is shown in Fig. 5.22. It can be seen that the high details in the original frame result in more fuzziness in the reconstructed frame. This supports the initial observation that the UVQ technique does not reproduce the edges very well [66]. The corresponding results are obtained using the MPEG-1 coder and shown in Fig. 5.10. For the Miss America sequence, the pyramid coder and the MPEG-1 coder performs comparable in terms of the PSNR value. The reconstructed and enhanced error frames for the frame number 4 of the

Miss America sequence are shown in Fig. 5.17 and 5.18 for subjective evaluation. It can be seen that the transform coding has overall lower fuzziness than the UVQ coding resulting in better subjective quality, especially at the sharp edges. For the Salesman sequence, MPEG-1 outperforms the pyramidal coder by 1 to 2 db PSNR values. The main reason for the lower performance of the pyramidal coder is that the Salesman sequence has very high details and the UVQ technique in general fails to reproduce high details well resulting in degradations. Now the interesting observation is that at the scene change, the performance of the pyramidal coder is better than the MPEG-1 coder. Pyramidal coding results in constant PSNR values for the Salesman sequence but the MPEG-1 coder has up to 3 db lower performance than the PSNR values after the scene change (Fig. 5.10).

The fourth set of experiments was performed for lossless coding. The encoding errors are delivered to the lower levels of the pyramid using error delivery algorithms presented in section 5.2. The final level of the pyramid is encoded using the Huffman coding scheme. For the first 8 frames of the Salesman sequence, the lossless bit rate obtained is around 4.1 bits/pixel per frame and for the Miss America sequence it is 3.42 bits/pixel. Thus, a lossless compression factor of around 2:1 is obtained even for the high detail Salesman sequence.

In the next set of experiments, the target bit rate was fixed at 1.5 Mb/s for the proposed video coder. The iterative method discussed in section 5.3 was used in combination with a bit allocation method. The values of α_1 and α_2 were taken as 2. The percentage bit rate variation around the target bit rate is plotted in Fig. 5.11. It can be seen that the variation is less than 0.5 % of the target bit rate.

In the last set of experiments, the performance of the proposed pyramidal video codec is tested for HDTV video compression. The Calendar sequence at CCIR resolution (720×480) is used for simulations. The Calendar sequence as shown in Fig. 5.23 has very high spatial details with a first order entropy of 7.4 bits/pixel. The

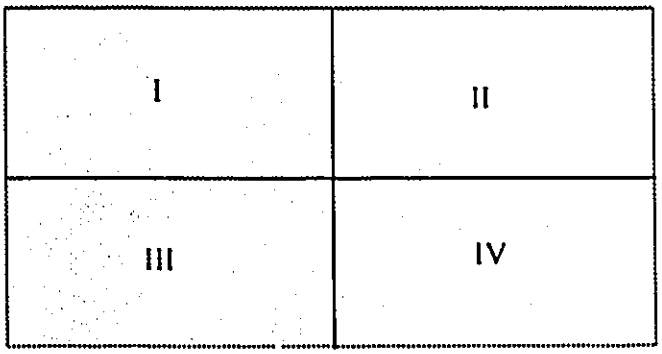

Figure 5.5: Partition of the large HDTV image into four subimages

basic motivation for this simulation is to measure the performance of the coder at very high bit rates for high quality images. Note that the MPEG-2 video compression standard is expected to address this problem.

In this simulation, each input frame is first divided into set of four subimages as shown in Fig. 5.5. The adaptive pyramid is built on each subimages resulting in four pyramids for one GOP of the video sequence. Advantage is taken of the fact that the higher spatial resolution may result in different amounts of temporal and spatial activity in different parts of the frame. Bidirectional motion estimation deals with the problems of the covered/uncovered areas and hence the division into four subimages does not degrade the performance of the codec.

Promising simulation results have been obtained for the Calendar sequence. The reconstructed and enhanced error frames are shown in Figs. 5.24 and 5.25 at bit rate of 6 Mb/s which corresponds to 0.5 bits/pixel. It can be seen that most of the details of the scene is preserved after coding. Some fuzziness can be seen at very sharp edges such as ball, fine numbers, etc.

5.5 Features of the Pyramidal Coder

The proposed video coder can support a number of general video coding features. Random access of the frames is possible by accessing the top level of the pyramid. Hence, this makes the compressed video an addressable-video. Fast forward and reverse play-back can be achieved in the encoded video. If we use fixed bits per pyramid (GOP), then video editing is feasible. The pyramidal video encoder is robust to errors such as coding error accumulation and packet loss in the channel. For robustness to channel errors, it is required that the upper levels of the pyramid be transmitted through a more secure channel. Format flexibility is feasible in the video by properly choosing the right pyramid depending on the application. This makes the proposed video coder a generic coder (application independent).

5.6 Conclusions

In this chapter, a video coder based on the 3D adaptive pyramid structure is proposed. The coder exploits spatio-temporal correlation using the 3D adaptive pyramid data structure and spatial correlation using an intra-frame coder (UVQ). The lossy coder employed to code the upper levels of the pyramid results in an accumulation of the errors. These coding errors are delivered to the lower levels of the pyramid. Algorithms are described for temporal and spatial error delivery for the prediction difference pyramid. A hierarchical buffer control scheme which uses an iterative technique to achieve the target bit rate in combination with a bit allocation method is presented. Detailed simulation results are presented for both video conferencing as well as HDTV applications. The performance and structure of the proposed pyramidal video coder is compared with those of the MPEG video compression standard. Finally, the main features of the pyramidal codec are summarized.

DPCM Motion Vectors	*VL Codes*
-11 & 11	0000 0100 011
-10 & 12	0000 0100 11
-9 & 13	0000 0101 01
-8 & 14	0000 0101 11
-7 & 15	0000 0111
-6 & 16	0000 1001
-5 & 17	0000 1011
-4 & 18	0000 111
-3 & 19	0001 1
-2 & 20	0011
-1	011
0	1
1	010
2 & -20	0010
3 & -19	0001 0
4 & -18	0000 110
5 & -17	0000 1010
6 & -16	0000 1000
7 & -15	0000 0110
8 & -14	0000 0101 10
9 & -13	0000 0101 00
10 & -12	0000 0100 10

Table 5.2: Variable length codes for the DPCM motion vectors

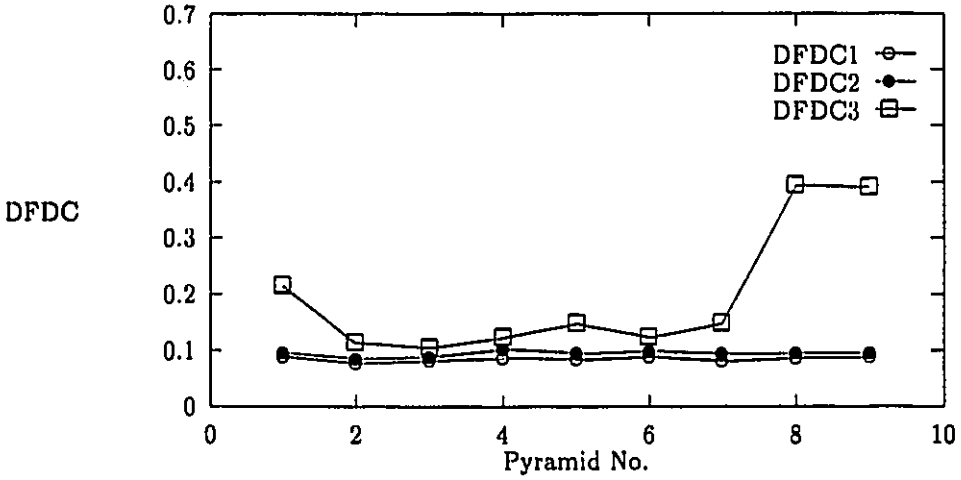

Figure 5.6: DFDC of motion estimation at different pyramid levels for the Miss America sequence, DFDC1: Bidirectional at level 1, DFDC2: Bidirectional at level 2, DFDC3: Unidirectional at level 3

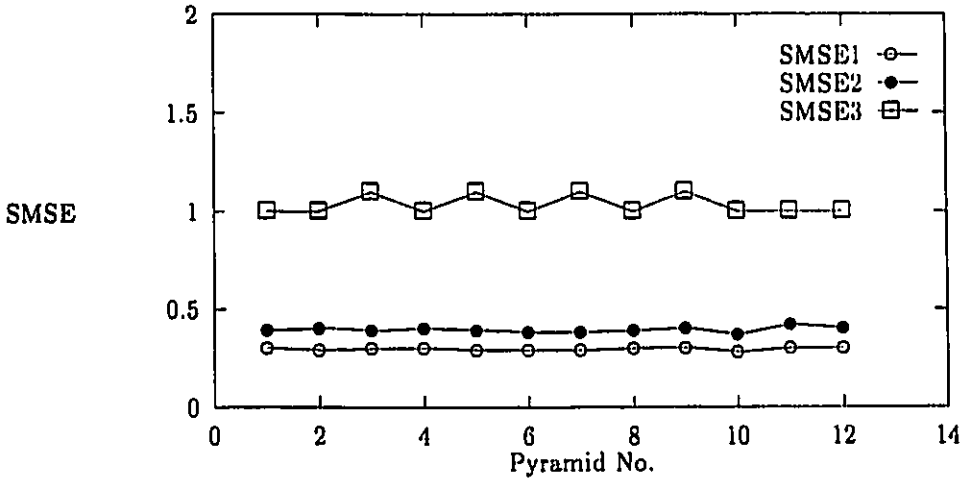

Figure 5.7: Spatial difference pyramid normalized (by 100) MSE, SMSE, for the Miss America sequence at different pyramid levels, SMSE1: Level 1, SMSE2: Level 2, SMSE3: Level 3

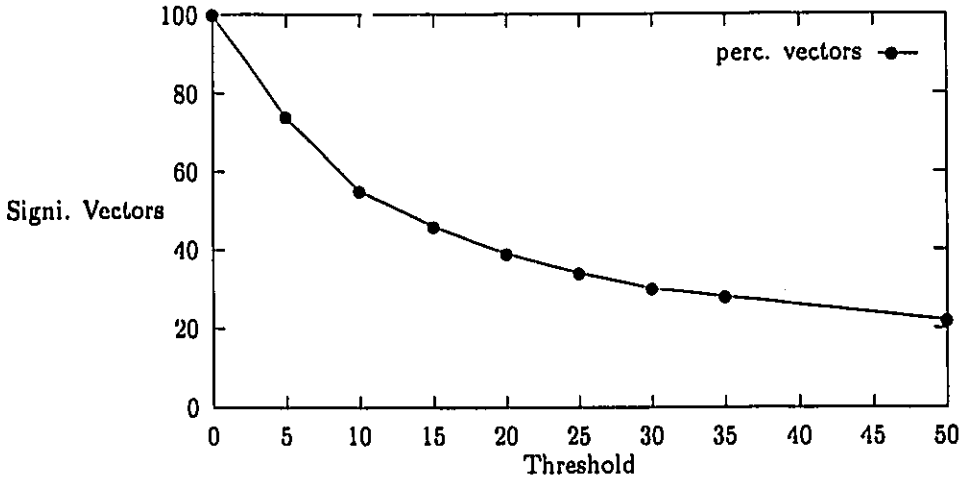

Figure 5.8: Block classification threshold vs. percentage significant vectors for the Miss America sequence

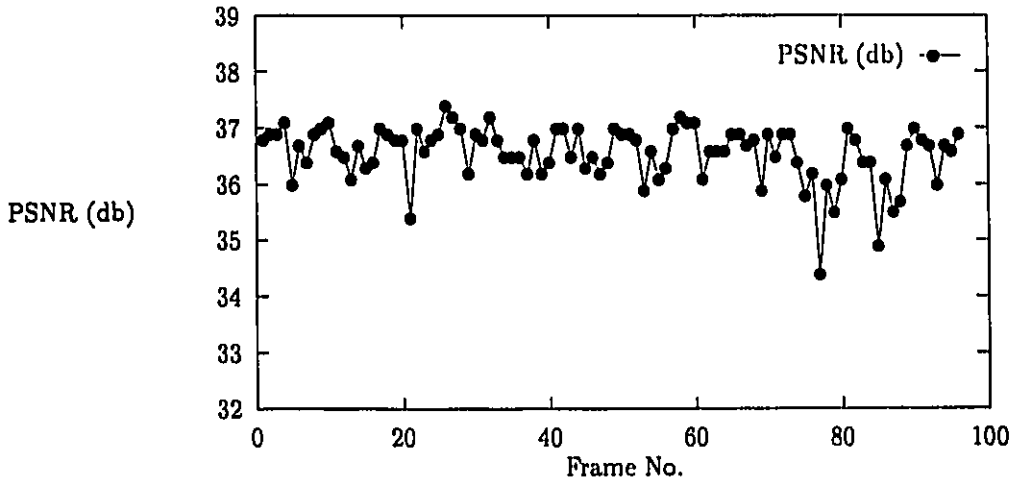

Figure 5.9: PSNR vs. frame no. for 96 frames of the Miss America sequence

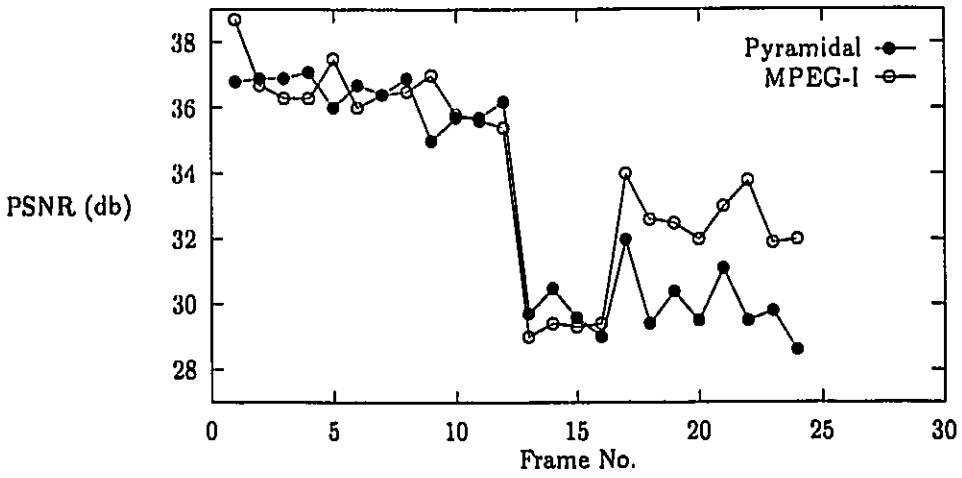

Figure 5.10: PSNR vs. frame no. for a scene change after frame no. 12 from the Miss America seq. to the Salesman seq. (Bit rate 1.5 Mb/s)

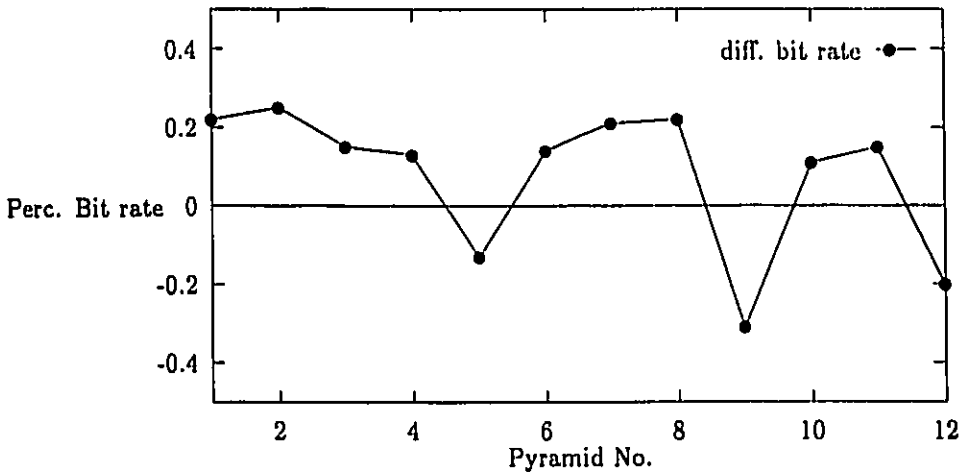

Figure 5.11: Bit rate variation with respect to the target bit rate of 1.5 Mb/s for the Miss America sequence

Figure 5.12: Original frame no. 4 of the Miss America sequence

Figure 5.13: Reconstructed frame no. 4 of the Miss America sequence using the adaptive temporal/spatial pyramid at 1.5 Mb/s

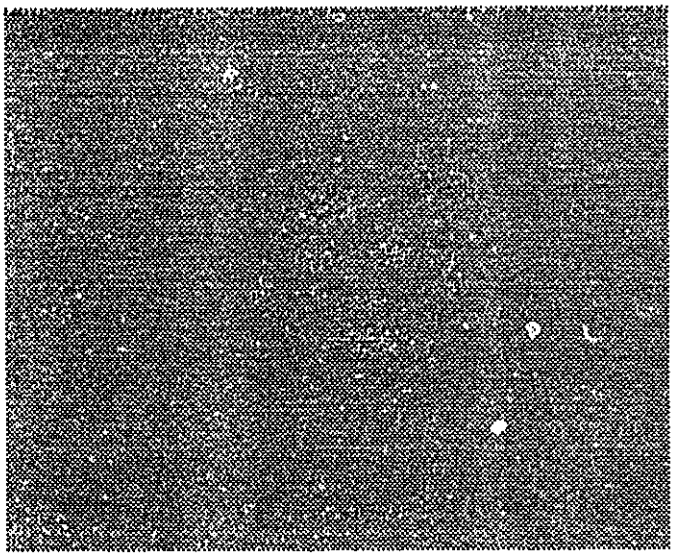

Figure 5.14: Error in the reconstructed frame no. 4 of the Miss America sequence using the adaptive temporal/spatial pyramid at 1.5 Mb/s

Figure 5.15: Reconstructed frame no. 4 of the Miss America sequence using the spatial pyramid at 1.5 Mb/s

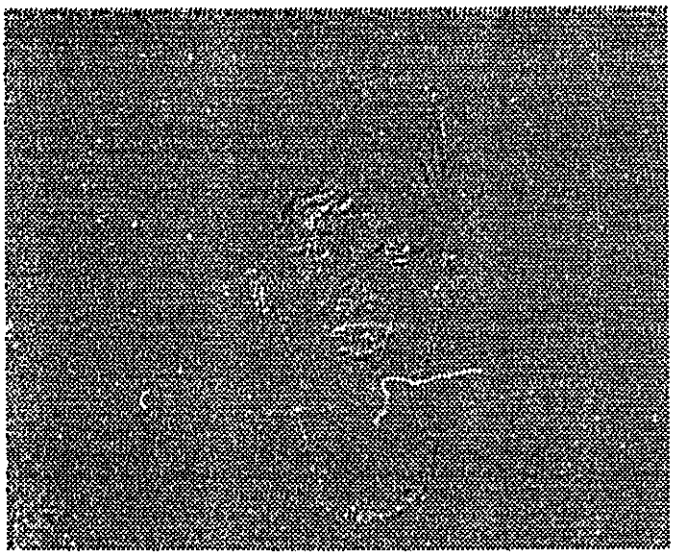

Figure 5.16: Error in the reconstructed frame no. 4 of the Miss America sequence using the spatial pyramid at 1.5 Mb/s

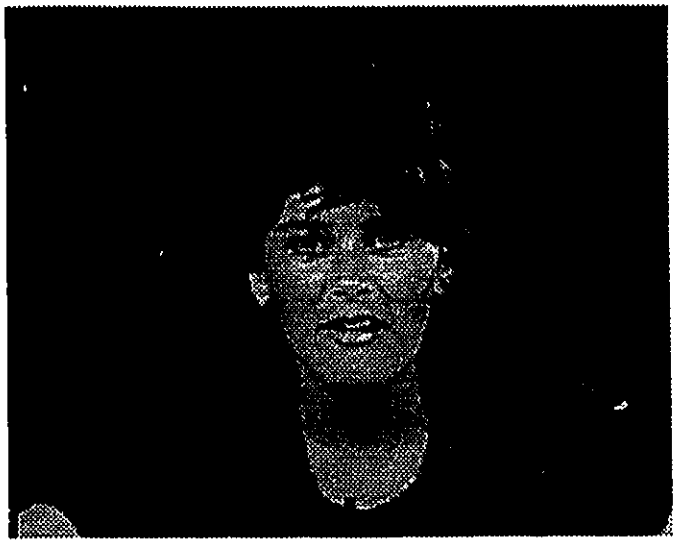

Figure 5.17: Reconstructed frame no. 4 of the Miss America sequence using the MPEG coder at 1.5 Mb/s

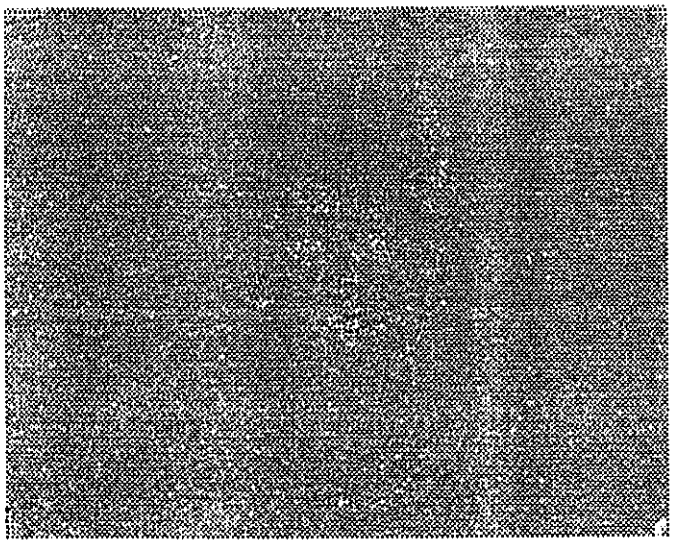

Figure 5.18: Error in the reconstructed frame no. 4 of the Miss America sequence using the MPEG coder at 1.5 Mb/s

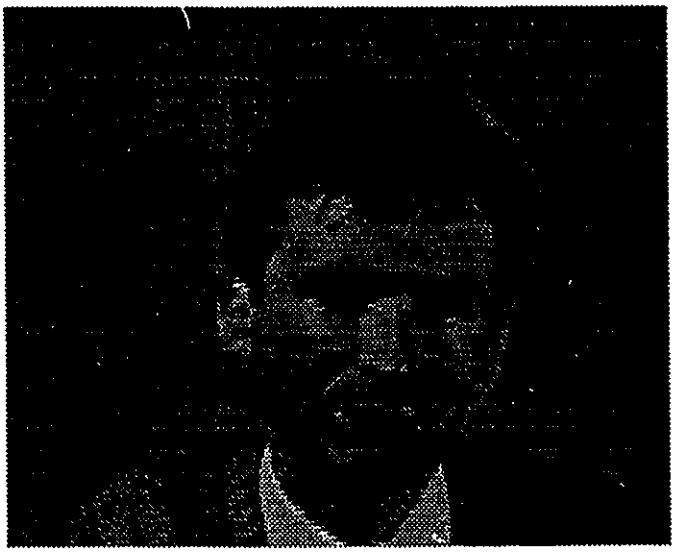

Figure 5.19: Original frame no. 2 of the Albert sequence (Used in the training Sequence to design universal codebooks)

Figure 5.20: Original frame no. 2 of the Salesman sequence (Used in the training Sequence to design universal codebooks)

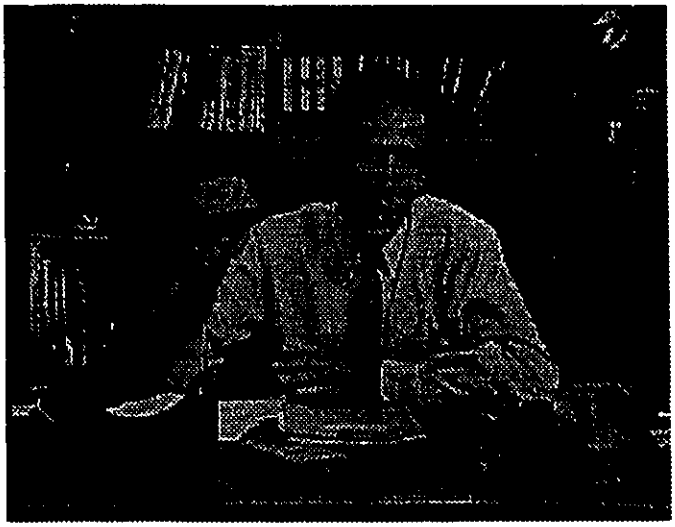

Figure 5.21: Reconstructed frame no. 2 of the Salesman sequence after the scene change from the Miss America sequence at 1.5 Mb/s

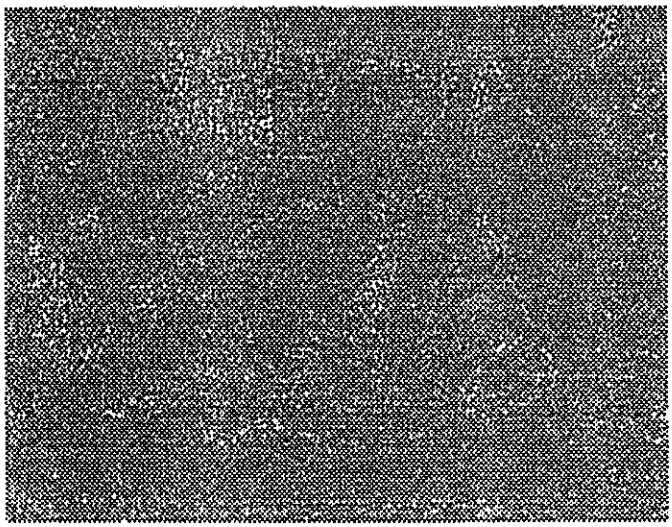

Figure 5.22: Error in the reconstructed frame no. 4 of the Salesman sequence at 1.5 Mb/s

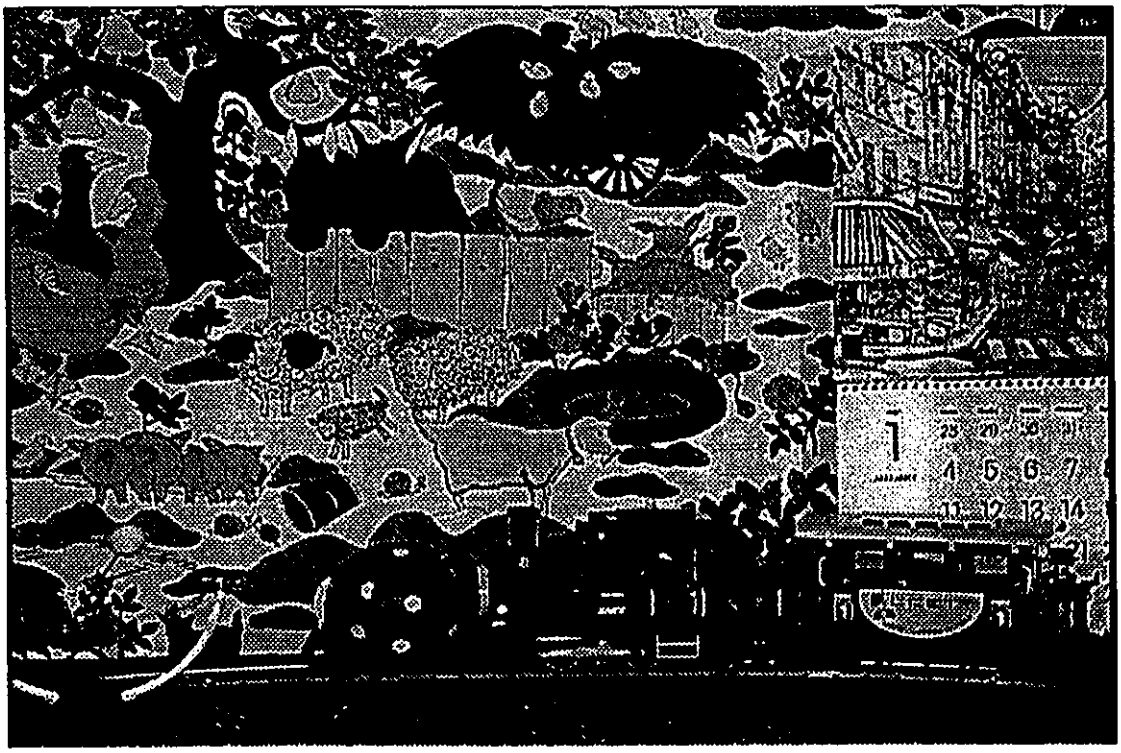

Figure 5.23: Original frame no. 8 of the Calendar sequence

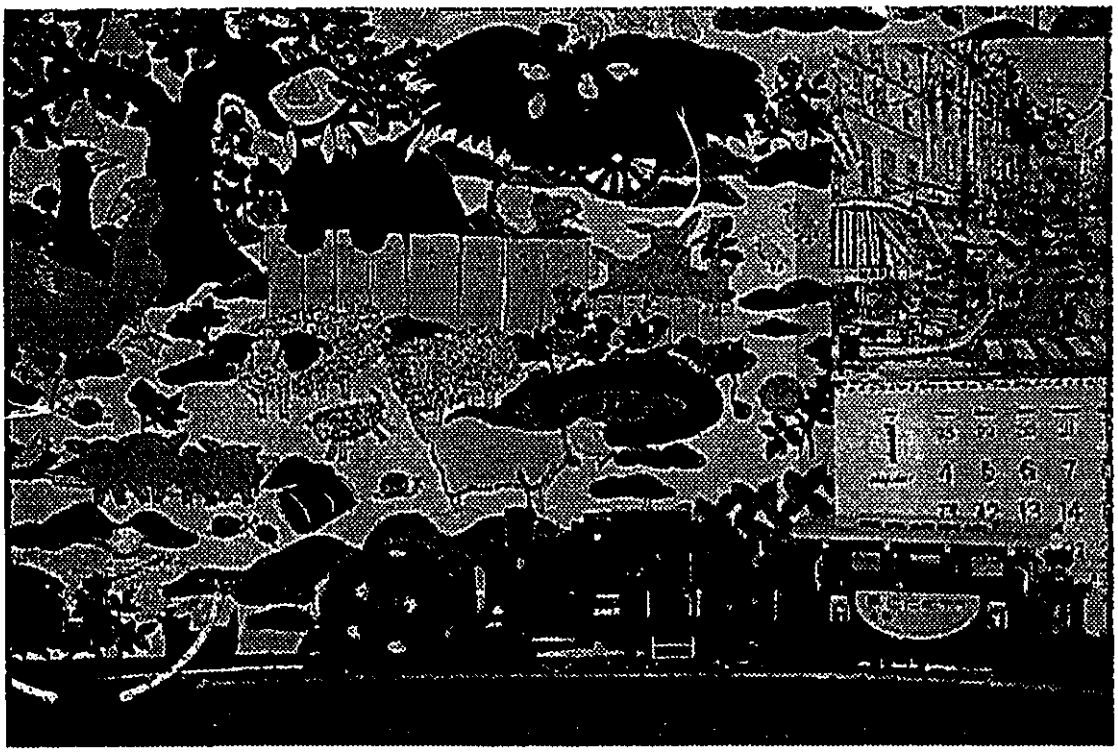

Figure 5.24: Reconstructed frame no. 8 of the Calendar sequence at 6.0 Mb/s

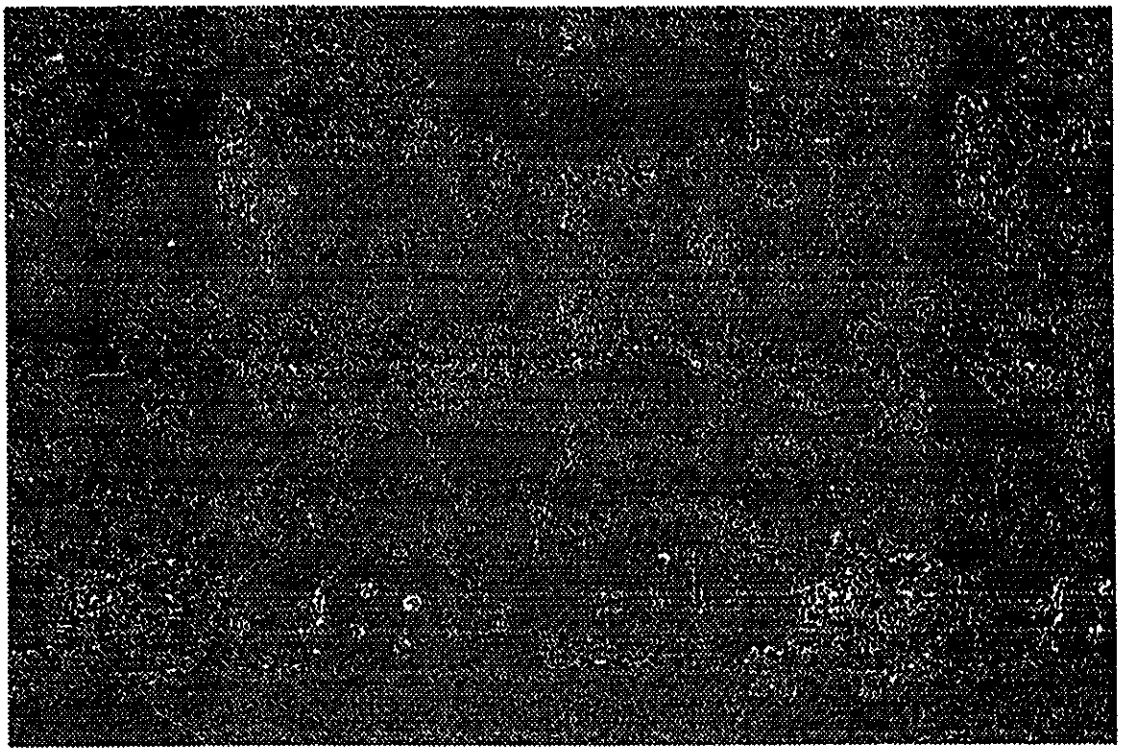

Figure 5.25: Error in the reconstructed frame no. 8 of the Calendar sequence at 6.0 Mb/s

Chapter 6

Conclusions and Future work

6.1 Conclusions

In this thesis, we have evaluated the performance of different temporal/spatial pyramid data structures for video compression in terms of the first order entropy. Based on this study, we propose an efficient 3D adaptive temporal/spatial pyramid data structure for video compression. The 3D adaptive temporal/spatial prediction difference pyramid achieves the lowest first order entropy. The adaptive pyramid exploits temporal vs. spatial correlation present in the video signal more efficiently and performs well even during the scene changes.

A video codec based on the 3D adaptive pyramid was presented. The video coder not only exploits the temporal and spatial correlation as in the MPEG coder but also exploits the spatio-temporal correlation using the 3D adaptive pyramid. Adaptivity in the pyramid is achieved by using a control signal derived from motion estimation. The 3D pyramid is encoded using the intra-frame universal vector quantization technique. The vectors are classified into significant/non-significant vectors before encoding and the non-significant vectors are discarded resulting in more efficient use of the bit rate. A hierarchical buffer control scheme is employed in combination with a bit allocation method which assigns bits to different levels of the pyramid depending on the number of I, P and B frames. Encoding errors introduced at the upper levels are fed-forward to the lower levels using temporal and spatial error delivery schemes which makes lossless coding possible. Simulation results indicate that the lossless compression factor of 2:1 is achieved even for the high detail sequence of the Salesman. Excellent subjective quality as well as objective quality are obtained at a bit rate of 1.5 Mb/s for the CCITT test sequences. For CCIR resolution high detail sequences, very good subjective quality can be obtained at a bit rate of 6 Mb/s. Furthermore, smooth transition is achieved in the case of scene changes without sacrificing picture quality. In addition, the algorithm is well suited for constant quality, constant bit rate applications with simple buffer control.

The proposed video coder has a number of similarities with the MPEG standard and can be implemented using the MPEG structure with minor modifications. Both the structures use bidirectional motion estimation and the concepts of I, P and B frames. The 3D adaptive pyramid switches to the spatial pyramid in case of lower temporal correlation in the video signal whereas MPEG coder does not switch to the spatial pyramid. Finally, the proposed video coder exhibits a number of useful features which are supported by MPEG.

6.2 Possible Extension of the Work

Some modifications to the algorithm can be made to improve the performance of the proposed coder. For example, a smart technique to form the 3D adaptive pyramid that exploits the local nature of the frames is an interesting area of research. The non-significant vectors in the universal codebooks can be discarded resulting in further reductions in bit rate. The encoding flow as proposed for MPEG-I coder may give superior performance. Quadtree segmentation can be combined with UVQ to achieve better compression ratios by exploiting the homogeneous regions of the error frames effectively. In addition, classified vector quantization (CVQ) can be employed to obtain the higher subjective quality of the images. In CVQ, the edges are reproduced very well and overall fuzziness is reduced. Smart encoding techniques such as a combination of VQ and VLDCT may give better performance. Here, the coder recognizes different areas in the image/video and employs suitable coding technique.

The extension of the algorithm to color video sequences is a possible research direction where higher compression ratios can be achieved. Performance of the algorithm can be studied for packet video transmission over noisy channels. For the digital TV application, better approach should be employed to efficiently exploit the higher spatial correlation in the sequences than the one used in this thesis. Study of

various bit streams for the proposed video coder to obtain the optimized bit stream that achieves the best performance for the given application is an area of considerable interest. Scalable video transmission using 3D adaptive pyramid is a possible research direction where the video signal is transmitted at different resolution scales. Real-time encoding using the proposed video coder is not possible. Some fast techniques can be employed especially for motion estimation to obtain higher speed-up. Simulation experiments were performed using the progressive scan sequences. This can be extended to the interlace video where higher compression ratio may be possible because of the lower spatial resolution. A unified software platform can be developed to evaluate different video coding schemes. Finally, some architectural work in terms of the hardware can be done for possible VLSI implementation of the codec.

Bibliography

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

[2] A. K. Jain, *Fundamentals of Digital Image Processing,* Prentice-Hall, Englewood Cliffs, New Jersey, 1989.

[3] N. Jayant, "Signal Compression: Technology Targets and Research Directions," *IEEE Journal on Selected Areas in Communications,* vol. 10, no. 5, pp. 796-818, June 1992.

[4] S. Singhal, D. Le Gall, C. T. Chen, "Source Coding of Speech and Video Signals," *Proceedings of the IEEE,* vol. 78, no. 7, pp. 1233-1249, July 1992.

[5] R. Forchheimer, T. Kronander, "Image Coding-From Waveforms to Animation," *IEEE Transactions on Acoustics, Speech and Signal Processing,* vol. 37, no. 12, pp. 2008-2023, December 1989.

[6] A. Habibi, "Survey of Adaptive Image Coding Techniques," *IEEE Transactions on Communications,* vol. COM-25, no. 11, pp. 1275-1284, November 1977.

[7] E. Dubois, "The Sampling and Reconstruction of Time-Varying Imagery with Application in Video Systems," *Proceedings of the IEEE,* vol. 73, no. 4, pp. 502-522, April 1985.

[8] H. Musmann, P. Pirsch, H. J. Grallert, "Advances in Picture Coding," *Proceedings of the IEEE,* vol. 73, no. 4, pp. 523-548, April 1985.

[9] M. Kunt, A. Ikonomopoulos, M. Kocher, "Second-Generation Image Coding Techniques," *Proceedings of the IEEE,* vol. 73, no. 4, pp. 549-574, April 1985.

[10] A. K. Jain, "Image Data Compression: A Review," *Proceedings of the IEEE,* vol. 69, no. 3, pp. 349-389, March 1981.

[11] L. D. Davisson, "Rate-Distortion Theory and Application," *Proceedings of the IEEE,* vol. 60, no. 7, pp. 800-808, July 1972.

[12] D. A. Huffman, "A Method for the Construction of Minimum-redundancy Codes," *Proceedings of the I.R.E,* pp.1098-1101, September 1952.

[13] I. H. Witten, R. M. Neal, J. G. Cleary, "Arithmetic Coding for Data Compression," *Communications of the ACM,* vol. 30, no. 6, pp. 520-540, June 1987.

[14] J. He, E. L. Dereniak, "Error-free Image Compression algorithm using Classifying-Sequencing Techniques," *Applied Optics,* vol. 31, no. 14, pp. 2554-2559, May 1992.

[15] G. K. Wallace, "The JPEG Still Picture Compression Standard," *Communication of the ACM,* vol. 34, no. 4, pp. 30-45, April 1991.

[16] R. K. Jurgen, "Digital Video," *IEEE Spectrum Magazine,* pp. 24-30, March 1992.

[17] P. H. Ang, P. A. Ruetz and D. Auld, "Video Compression makes big gains," *IEEE Spectrum Magazine,* pp. 16-19, October 1991.

[18] "Draft Revision of Recommendations H.261: Video Codec for Audiovisual services at p x 64 Kbit/s," *Signal Processing: Image Communication 2,* vol. 2, no. 2, pp. 221-239, August 1990.

[19] M. Liou, "Overview of the px64 Video Coding Standard," *Communications of the ACM,* vol. 34, no. 4, pp. 59-63, April 1991.

[20] D. Le Gall, "MPEG: A Video Compression Standard for Multimedia Applications," *Communication of the ACM,* vol. 34, no. 4, pp. 46-58, April 1991.

[21] K. Patel, B. Smith, L. Rowe, "Performance of a Software MPEG Video Decoder," Anonymous FTP to *toe.cs.berkeley.edu,* directory /pub/multimedia/mpeg/ 1993.

[22] Draft, "Coding of Moving Pictures and Associated Audio," *MPEG Video CD Editorial Committee,* MPEG 90/176 Rev. 2, December 1989.

[23] M. Liou, "Visual Telephony as an ISDN Application," *IEEE Communications Magazine*, pp. 30-37, February 1990.

[24] K. Konstantinides, V. Bhaskaran, "Monolithic Architectures for Image Processing and Compression," *IEEE Computer Graphics and Applications Magazine*, pp. 75-86, November 1992.

[25] Data Sheets, "CCITT Video Compression Chipset," *LSI Logic Corporation*, Milipitas, California.

[26] Q. Wang, R. J. Clarke, "Motion Estimation and Compensation for Image Sequence Coding," *Signal Processing: Image Communication 4*, pp. 161-174, April 1992.

[27] J. R. Jain, 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.

[28] F. Glazer, "Scene Matching by Hierarchical Correlation," *IEEE Conference Proceedings*, pp. 432-441, 1983.

[29] M. Bierling, R. Thoma, "Motion Compensating Field Interpolation using a Hierarchically Structured Displaced Estimator," *Signal Processing 11*, North-Holland, pp. 387-404, 1986.

[30] C. Bergeron, E. Dubois, "Gradient-Based Algorithms for Block-Oriented MAP Estimation of Motion and Application to Motion-Compensated Temporal Interpolation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 1, pp. 72-85, 1991.

[31] A. Puri, H. Hang, D. Schilling, "Interframe Coding with Variable Block-size Motion Compensation," *Proceedings of GLOBECOM*, pp. 65-69, 1987.

[32] J. Moon, J. Koh, S. Kim, J. Kim, "Spatial-Temporal Prediction Algorithm with Recursive Motion Compensation," *Proceedings of GLOBECOM*, pp. 70-72, 1987.

[33] R. J. Clarke, "On Transform Coding of Motion-Compensated Difference Images," *IEE Proceedings-I*, vol. 139, no. 3, pp. 372-376, June 1992.

[34] L. Wang, "Bit Rate Control for Hybrid DPCM/DCT Video Codec," *Lab. Report*, 1992, Communications Research Center, Ottawa, Canada.

[35] L. Wang, "A Simple Approach to Bit Rate Control for Video Compression," *Proceedings of The Canadian Conference of Electrical and Computer Engineers*, pp. TA4.20.1-4, 1992.

[36] C. Horne, A. Puri, "Video Coding with Adaptive Quantization and Rate Control," *SPIE Visual Communications and Image Processing*, vol. 1818, pp. 798-806, 1992.

[37] T. Mochizuki, J. Ohki, "Feedforward Control of Hybrid Transform/Predictive Coder," *SPIE Visual Communications and Image Processing*, vol. 1001, pp. 826-833, 1988.

[38] K. Xie, L. Eycken, A. Oosterlinck, "Motion-compensated Adaptive Inter/Intra Frame Prediction," *SPIE Visual Communications and Image Processing*, vol. 1360, pp. 1798-1809, 1990.

[39] H. Yamamoto, Y. Hatori, H. Murakami, "30 Mbit/s Codec for the NTSC Color TV Signal Using an Interfield-Intrafield Adaptive Prediction," *IEEE Transactions on Communications*, vol. COM-29, no. 12, pp. 1859-1867, December 1981.

[40] F. A. Kamangar, K. R. Rao, "Interfield Hybrid Coding of Component Color Television Signals," *IEEE Transactions on Communications*, vol. COM-29, no. 12, pp. 1740-1753, December 1981.

[41] R. Picco, F. Bellifemine, A. Chimienti, "Analysis of a 2D-DCT Image Coding Scheme with Motion Compensation and Vector Quantization," *SPIE Visual Communications and Image Processing*, vol. 1810, pp. 1810-1821, 1990.

[42] J. A. Roese, W. K. Pratt, G. S. Robinson, "Interframe Cosine Transform Image Coding," *IEEE Transactions on Communications*, vol. COM-25, no. 11, pp. 1329-1339, 1977.

[43] W. Chen, W. K. Pratt, "Scene Adaptive Coder," *IEEE Transactions on Communications*, vol. COM-32, no. 3, pp. 225-232, 1984.

[44] W. Chen, H. Smith, "Adaptive Coding of Monochrome and Color Images," *IEEE Transactions on Communications*, vol. COM-25, no. 11, pp. 1285-1292, 1977.

[45] J. W. Woods, "Subband Coding of Images," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-34, no. 5, pp. 1278-1288, October 1986.

[46] P. H. Westerink, J. Biemond, D. E. Boekee, "Progressive Transmission of Images using Subband Coding," *IEEE Conference on Acoustics, Speech and Signal Processing*, pp. 1811-1814, 1989.

[47] F. Bosveld, R. L. Lagendijk, J. Biemond, "3D Subband Decomposions for Hierarchical Video Coding," *SPIE Visual Communications and Image Processing*, vol. 1605, pp. 769-780, 1991.

[48] J. R. Ohm, "Temporal Domain Subband Video Coding with Motion Compensation," *IEEE Conference on Acoustics, Speech and Signal Processing*, pp. 229-232, 1992.

[49] U. Golz, R. Schafer, "Considerations on the Possibility to exchange Temporal Against Spatial Resolution in Image Coding," *Signal Processing: Image Communication 2*, pp. 39-51, 1990.

[50] J. Biemond, P. H. Westerink, F. Muller, "Subband Coding of image sequences at low bit rates," *SPIE Visual Communications and Image Processing*, vol. 1199, pp. 741-751, 1989.

[51] F. Bosveld, R. L. Lagendijk, J. Biemond, "Hierarchical Coding of HDTV," *Signal Processing: Image Communication 4*, pp. 195-225, 1992.

[52] L. Vandendorpe, "Hierarchical Transform and Subband Coding of Video Signals," *Signal Processing: Image Communication 4*, pp. 245-262, 1992.

[53] P. Strobach, "Tree-Structured Scene Adaptive Coder," *IEEE Transactions on Communications*, vol. COM-38, no. 4, pp. 477-486, April 1990.

[54] P. J. Burt, E. H. Adelson, "The Laplacian Pyramid as a Compact Image Code," *IEEE Transactions on Communications*, vol. COM-31, no. 4, pp. 532-540, April 1983.

[55] A. Sanz, C. Munoz, N. Garcia, "Hierarchical Predictive Approach to Image Coding," *IEEE Conference on Acoustics, Speech and Signal Processing*, pp. 113-116, 1985.

[56] M. Goldberg, L. Wang, "Comparative Performance of Pyramid Data Structures for Progressive Image Transmission," *IEEE Transactions on Communications*, vol. COM-39, no. 4, pp. 540-548, April 1991.

[57] L. Wang, M. Goldberg, "Progressive Image Transmission using Vector Quantization on Images in Pyramid Form," *IEEE Transactions on Communications*, vol. COM-37, no. 12, pp. 1339-1349, December 1989.

[58] L. Wang, M. Goldberg, "Reduced-difference Pyramid: A Data Structure for Progressive Image Transmission," *Optical Engineering*, vol. 28(7), pp. 708-716, July 1989.

[59] K. M. Uz, M. Vetterli, D. J. LeGall, "Interpolative Multiresolution Coding of Advanced Television with Compatible Subchannels," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 1, no. 1, pp. 86-99, March 1991.

[60] C. Stiller, D. Lappe, "Laplacian Pyramid Coding of Prediction Error Images," *SPIE Visual Communications and Image Processing*, vol. 1605, pp. 47-57, 1991.

[61] S. Sallent, L. Torres, L. Gils, "Three-Dimensional Adaptive Laplacian Pyramid Image Coding," *SPIE Visual Communications and Image Processing*, vol. 1360, pp. 627-638, 1990.

[62] T. Sikora, T. K. Tan, K. K. Pang, "A Two Layer Pyramid Image Coding Scheme for Interworking of Video Services in ATM," *SPIE Visual Communications and Image Processing*, vol. 1605, pp. 624-634, 1991.

[63] R. M. Gray, "Vector Quantization," *IEEE Acoustics, Speech and Signal Processing Magazine*, pp. 4-29, April 1984.

[64] Y. Linde, A. Buzo, R. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Transactions on Communications*, pp. 84-95, January 1980.

[65] N. M. Nasrabadi, R. A. King, "Image Coding Using Vector Quantization: A Review," *IEEE Transactions on Communications*, vol. 36, no. 8, pp. 957-971, August 1988.

[66] A. Gersho, B. Ramamurthi, "Image Coding Using Vector Quantization," *IEEE Conference on Acoustics, Speech and Signal Processing*, pp. 428-431, 1982.

[67] B. Ramamurthi, A. Gersho, "Classified Vector Quantization of Images," *IEEE Transactions on Communications*, vol. COM-34, no. 11, pp. 1105-1115, 1986.

[68] J. Kim, S. Lee, "A Transform Domain Classified Vector Quantizer for Image coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 2, no. 1, pp. 3-14, 1992.

[69] R. Aravind, A. Gersho, "Low-rate Image Coding with Finite-state Vector Quantization," *IEEE Conference on Acoustics, Speech and Signal Processing*, pp. 4.3.1-4.3.4, 1986.

[70] M. Goldberg, P. Boucher, S. Shlien, "Image Compression Using Adaptive Vector Quantization," *IEEE Transactions on Communications*, vol. COM-34, pp. 180-187, February 1986.

[71] S. Panchanathan, M. Goldberg, "Minimax Algorithm for Image Adaptive Vector Quantization," *IEE Proceedings-I*, vol. 138, no. 1, pp. 53-60, February 1991.

[72] M. Goldberg, H. Sun, "Image Sequence Coding Using Vector Quantization," *IEEE Transactions on Communications*, vol. COM-34, pp. 703-710, July 1986.

[73] T. Murakami, K. Asai, E. Yamazaki, "Vector Quantizer of Video Signals," *Electronic Letters*, vol. 7, pp. 1005-1006, November 1982.

[74] N. M. Nasrabadi, "Interframe Hierarchical Address-Vector Quantization," *Visual Communications and Image Processing*, vol. 1360, pp. 558-574, 1990.

[75] W. T. Chen, R. Chang, J. Wang, "Image Sequence Coding Using Adaptive Finite-State Vector Quantization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 2, no. 1, pp. 15-24, March 1992.

[76] H. Chen, Y. Chen, W. Hsu, "Low-rate Sequence Image Coding Via Vector Quantization," *Signal Processing 26*, pp. 265-283, 1992.

[77] J. Huguet, L. Torres, "Vector Quantization in Image Sequence Coding," *Signal Processing V: Theories and Applications*, pp. 1079-1082, 1990.

[78] L. Lu, W. A. Pearlman, "Multirate Image Sequence Coding with Quadtree Segmentation and Backward Motion Compensation," *Visual Communications and Image Processing*, vol. 1818, pp. 606-617, 1992.

[79] Y. Zhang, S. Zafar, "Motion-Compensated Wavelet Transform Coding for Color Video Compression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 2, no. 3, pp. 285-296, September, 1992.

[80] T. Ebrahimi, F. Dufaux, M. Kunt, "A Digital Video Codec for Medium Bitrate Transmission," *SPIE Visual Communications and Image Processing*, vol. 1605, pp. 2-15, 1991.

[81] T. Ebrahimi, "Perceptually Derived Localized Linear Operators - Application to Image Sequence Compression," *Ph.D. Thesis*, EPFL-Ecublens (DE), Switzerland, 1992.

[82] L. Wang, "Progressive Image Transmission," *Ph.D. Thesis*, University of Ottawa, 1988.

[83] E. A. Fox, "Advances in Interactive Digital Multimedia Systems," *IEEE Compter Magazine*, pp. 9-21, October 1991.

[84] R. Gandhi, L. Wang, S. Panchanathan, M. Goldberg, "An MPEG-like Pyramidal Video Coder," Accepted for presentation at the *SPIE Visual Communications and Image Processing*, Cambridge, November 1993.

[85] R. Gandhi, L. Wang, S. Panchanathan, M. Goldberg, "An MPEG-like Pyramidal Video Coder," Submitted for possible publication in the *IEEE Transactions on Image Processing*, Special Issue on Image Sequence Compression, July 1994.

[86] R. Gandhi, L. Wang, S. Panchanathan, M. Goldberg, "3D Adaptive Pyramid for Compatible HDTV Video Compression," Accepted for presentation at the *International Workshop on HDTV*, Ottawa, October 1993.