CARNEGIE MELLON UNIVERSITY

STATISTICAL MODELING FOR NETWORKED VIDEO: CODING OPTIMIZATION, ERROR CONCEALMENT AND TRAFFIC ANALYSIS

A DISSERTATION
SUBMITTED TO THE GRADUATE SCHOOL
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
for the degree

DOCTOR OF PHILOSOPHY
in
ELECTRICAL AND COMPUTER ENGINEERING

by

Deepak Srinivas Turaga

Pittsburgh, Pennsylvania
July, 2001

Abstract

Video coding has attracted much attention in the recent past, especially due to the large amount of digital video content available today. Video transmission and storage requirements result in efficient compression techniques with many different evolving compression standards, such as H.263 and MPEG-4. Besides efficient compression, video coding techniques have to ensure good video quality while involving real-time processing. Complexity, quality and bit rate are factors that measure success of a video coding scheme. The focus of this thesis is on optimizing the video coding process to improve performance in terms of one or more of these three factors. We use statistical modeling techniques to achieve this optimization goal. Specifically we examine two parts of the video coding process, mode decisions and error concealment. Mode decisions involve selecting the optimal modes of operation under certain constraints. We build a classification based framework for making mode decisions to minimize the coding cost that may be defined in terms of the three parameters, complexity, quality, and bit rate. We propose a scheme for model based error concealment, i.e., using a statistical model for the region of interest to replenish any data lost due to errors in network transmission. We introduce a new and efficient statistical model called Mixture of Principal Components (MPC) to capture the properties of the region of interest. We show that this model is more efficient than the traditional Principal Component Analysis (PCA) in capturing data variations, especially when the data consists of samples distributed in multiple clusters. We also use this model for an example face recognition task in order to highlight some other applications for this general statistical framework. We realize that both the mode decisions as well as the error concealment optimizations require feedback from the network regarding the available bandwidth, loss probability, and delay. Hence, in the last part of this thesis we focus on modeling the variable bit rate video traffic so that we may use this traffic to probe the network to determine the network condition and optimize our coding algorithms appropriately.

Acknowledgements

I would like to start by thanking my advisor Prof. Tsuhan Chen who has been a constant source of inspiration through the course of my PhD. I am very grateful to him for his continuous support, encouragement, and great enthusiasm. His insights have been very valuable in enabling my exploration of new research directions as well helping me overcome the many problems that we encountered. I have also benefited vastly from his emphasis on improving written and oral presentation skills. He has been a great teacher, motivator and research mentor. I have learnt many valuable lessons through our interaction and hope to carry forward his high standards through my career and life. I am fortunate to have had him as my advisor.

I would also like to thank Prof. Jose Moura, Prof. B. V. K. Vijaya Kumar and Dr. Thomas Gardos for their interest in my work, their patience, and the time they have invested in being part of my thesis committee. I am grateful to them for their many insights and comments that have led to the improvement of this body of work. I hope to continue my association with them through my career. I am fortunate to have been part of the Electrical and Computer Engineering department, where I have encountered many excellent faculty and students that have helped stimulate and challenge me, and made me enjoy the learning process.

My lab-mates Fu Jie Huang, Ta-Chien Lin, Howard Leung, Trista Chen, Xiaoming Liu and Cha Zhang have been of great help during the entire course of this PhD. They have not only contributed to my research through many suggestions and discussions, but have also enriched my stay here at CMU through their personal interaction.

I cannot but thank my roommates Amit Itagi, Deepak Sharma and Dhanunjay Boyalakuntla for the many happy times that we had during our stay together. I will carry these pleasant memories with me always. I hope they will too. It would be remiss of me not to mention the very vibrant Indian student community here that helped make Pittsburgh home away from home. Thanks for all the great times.

Finally, I have to acknowledge the support I received from my family, my brother Kiran, my father Ravi and my mother Sudha. Although I have been physically distant, they have always been a part of my being, always there for me. All that I have achieved is due to their constant support and love. Mere words cannot describe the love and appreciation I feel for them, and I hope to make them proud by living up to what they have taught me. I dedicate this thesis to my parents.

Table of Contents

List of Tables

List of Illustrations

1. Introduction

Video coding includes both the encoding from video data to a bitstream as well as the decoding from the bitstream to video data. The goal of video coding is to achieve efficient compression while maintaining as high a quality of the reconstructed video as possible. Besides the compression ratio and the quality, real time video encoding and decoding is a primary requirement of video coding systems. Hence, overall video coders have to perform as well as possible in terms of the "*complexity-quality-bit rate*" tradeoff. Complexity may be measured in terms of the time needed for processing the data. Quality consists of two parts, the spatial quality and the temporal quality and many spatio-temporal quality metrics have been defined that obtain an objective assessment of the overall quality of a video stream. The bit rate is measured in terms of the number of bits needed to code the sequence.

Video coding standards specify the bitstream syntax completely, but allow for optimizations in the encoding process, in terms of algorithms and mode decisions used, and any post-processing in the decoding process. These optimizations are allowed as coders designed for different applications have different requirements. For instance when the application requires real-time coding, the complexity of the coding becomes critical, while in applications with off-line coding, the quality and the coding efficiency are more critical than the complexity.

The focus of this thesis is on optimizing the video coding process to improve performance in terms of one or more parameters of the complexity-quality-bit rate tradeoff. Specifically in order to improve the performance, we target two different parts of the coding process, various *mode decisions* part of the encoding process, and *error concealment*, part of the decoding process. In addition, we also examine *modeling of video traffic* that is produced by the encoding process. This is because work on optimizing mode decisions and error concealment requires information such as transfer delay, transfer rate and loss probability, from the network. Using accurate models for network traffic enables us to estimate these parameters from networks. The recurrent theme underlying all of our work is the use of statistical modeling techniques to optimize the video coding process. Hence we build a classification based framework for making optimal mode decisions, a statistical modeling framework for capturing data variations for error concealment, and doubly stochastic models for capturing the properties of video traffic.

All these three parts of our thesis are closely related to one another and the interaction between them is shown in Figure 1.1.

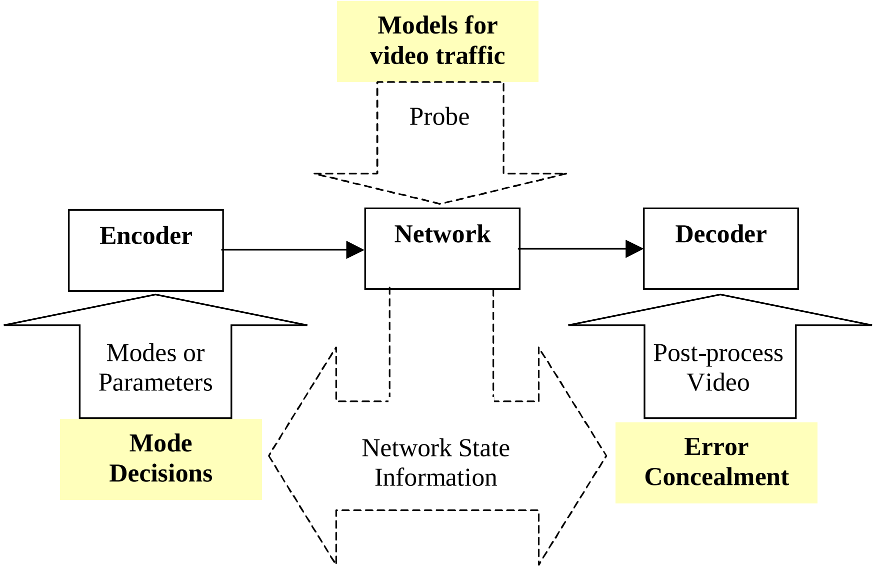

Figure 1.1. Interactions between different optimizations

In Figure 1.1, we show both the optimizations and modeling work in the context of the encoder, decoder and the network. The models for the traffic may be used to probe the network to obtain desired network state information, such as available bandwidth, loss probability, network delays, congestion etc. This information is very useful to network designers as it enables them to provide certain estimates and guarantees on the network performance. This information may also be used in the encoding and decoding. This information may be passed on to the mode decision optimizer, which selects the appropriate coding modes and parameters in order to modify the encoder output to meet these network constraints. This information may be also passed on to the error concealment module, which then adjusts the decoder post-processing to reduce the effects of the losses due to transmission over the network.

As an illustration we propose the following scenario. Using the models as a probe we determine the loss probability and delay associated with transmitting data at a certain rate over the network. This information can be fed into the mode decision optimizer that then changes the quantization step size, or decides to skip coding some frames in order to maximize the decoded video quality while meeting the constrained target rate. This information is also made available to the decoder, where the error concealment scheme can adjust the parameters to perform a more effective concealment. To demonstrate the performance of these optimizations and post-processing, we have implemented them in the H.263 framework, using the codec developed at

CMU [1]. These techniques are, however, not limited to one video coding standard, but may be applied to other standards as well.

1.1. Mode Decision Framework

Inherent in the coding process are many mode decisions that improve one or more of the parameters of the complexity-quality-bit rate tradeoff. These mode decisions may be at different levels of the coding process and may have different goals. Two examples of mode decisions are the Intra-Inter mode decision and the mode decision to skip or code a frame to solve the problem of rate control.

Inter coding a block is typically more efficient for scenes with simple backgrounds and moderate to low motion as a good prediction for the block may be found from the previous frame. Intra coding is more efficient for sequences with complex backgrounds and large, irregular motion, when a good prediction cannot be found for the current block. This Intra-Inter decision needs to be made for every block in the frame.

Rate control involves adjusting the coding parameters in order to meet a target bit rate provided by the network. The problem of rate control has been extensively studied in literature. Chen and Wong [2] and Choi and Park [3] try to solve the rate control problem using buffer control strategies. As against this, Hsu, Ortega and Reibman [4] propose algorithms for rate control using both source rate control as well as channel rate control over asynchronous transfer mode (ATM) networks. A majority of the work in these papers involves controlling the output rate of the encoder by changing the spatial quality. Spatial quality is controlled by the quantization step size, with a larger quantization step size leading to worse quality with correspondingly smaller bit rate. These strategies consider changing the frame rate for rate control as the last option, only when the target rate cannot be achieved by changing the quantization step size. Song, Kim and Kuo [5] propose rate control algorithms for low bit rate unconstrained VBR that allow for a variable target rate provided by the network. The algorithms they propose try to control the spatial and temporal quality simultaneously to improve the perceived quality. They use frame skipping or frame rate changes in order to improve the perceived quality and the primary algorithm to control the bit rate is through the use of the quantization step size. Work by Martins, Ding and Feig [6] tries to achieve rate control by combining the effect of coding and skipping a frame into a composite cost function. They, however lack granularity in their work, as the decision they make is not on a frame by frame basis. They make a decision for the future using information from coding the current frame. As an example, if after coding a frame they feel that they need to skip 10 frames, they do so without examining any of the following 10 frames.

Most of the work above uses models to estimate the relationship between bits needed to code frames and the quantization step size. This is useful to estimate the quantization step size needed to code the frame given a target rate. Some of these models were proposed by Chiang and Zhang [7] and by Ding and Liu [8]. A majority of existing rate control algorithms modify the quantization step size to control the rate of VBR encoders. Frame skipping to control the bit rate is used only as a last resort, i.e., when a buffer overflow occurs or when the buffer approaches overflow. Such algorithms do not consider the loss of quality that occurs by this arbitrary skipping of frames. This problem is more severe at low bit rates when it is difficult to achieve the target rate by changing the quantization step size alone and more frames need to be skipped.

We show that many of these mode decisions can be considered as binary hypothesis testing problems that are well understood in traditional classification theory. We develop a classification based framework for making mode decisions that are optimized in terms of minimizing the cost of coding, as measured in terms of one or more parameters of the coding process. We identify features that can be easily computed from the video data, and are good indicators of which mode would be optimal. We estimate probability density functions for these features under different hypotheses, corresponding to the different options in the mode decision. We then transform this minimization of cost to a more traditional error probability minimization problem and use standard classification techniques like the likelihood ratio test to solve it. We evaluate the performance of this classification approach using Intra-Inter mode decision and the frame coding versus skipping for rate control and compare the results with currently proposed mode decisions. We propose a classification based scheme for rate control that decides intelligently between changing the quantization step size and skipping the frame to achieve the desired target rate. We look at the suitability of skipping or coding a certain frame in terms of the impact it has on the rate as well as the impact it has on the quality before making a decision. We try to maximize perceived quality while achieving a fixed target rate.

1.2. Error Concealment

Loss of compressed video information during transmission leads to objectionable visual distortion in the reconstructed video. In order to minimize distortion at the decoder end many schemes for error resilience and error concealment have been developed. Error resilience includes schemes at the encoder end where some redundancy is introduced in the bitstream that makes it possible to recover lost information. Examples of these include error correcting codes, data partitioning techniques etc. Error concealment involves post-processing of the video at the decoder end to hide the effect of the transmission errors. Both error resilience and error concealment techniques may be used in conjunction to improve the quality of the decoded video. The focus of this thesis is on error concealment techniques. There is a lot of existing work on concealing errors due to losses incurred while transmitting data over networks or during data storage. Work in this domain includes work by Aign and Fazel [17], who use spatial domain interpolation, Sun and Kwok [18], who use projections onto convex sets (POCS) and Chen et al [19], who use temporal domain interpolation and overlapped block motion compensation. Atzori and De Natale [20] introduce a sketch-based approach to error concealment, where each frame is decomposed into a set of sketches, and error concealment involves recovering these sketches using some constraints on the sketch continuity, derivatives etc.

We propose a model based scheme for error concealment. We create a model for the region of interest, where we wish to conceal errors and use this model to replenish any missing data. We examine the principal component analysis (PCA) [9] as a model for our region of interest and find that it is not efficient in capturing data with large amounts of variation. Such a model based concealment approach is very useful especially for the MPEG-4 standard, which uses object based coding, thereby making it easy to determine regions of interest and build appropriate models for them.

PCA tries to model the data as a hyperplane embedded in the data space. Hence to represent large variations in the data set, PCA requires larger dimensional representation than would be required by a non-linear modeling technique. So many non-linear approaches to dimensionality reduction have been proposed. Hastie and Stuetzle [10] proposed principal surfaces as an alternative to PCA. This involves constructing parameterized surfaces through the data cluster to minimize the overall distance between the data points and the surfaces. Many neural network approximators for these principal surfaces of the high-dimensional data have also been proposed. Among these are the work by Oja [11] and by Kung and Diamantaras [12]. Other techniques such as Multi-Dimensional Scaling [13] that attempt to preserve the pair wise distances between data points during the dimensionality reduction have been proposed. Recently, other approaches such as Locally Linear Embedding (LLE) [14] have also been proposed. LLE computes low-dimensional, neighborhood preserving embeddings of the data by exploiting the local symmetries of linear reconstructions. Each data vector is first approximated as a weighted linear combination of its neighbors, and the optimal set of weights for least squared reconstruction error are determined. Then the data is transformed to lower dimension while keeping the same set of weights for each data vector. This way the local neighborhood relationships are preserved.

However, there are many disadvantages to these non-linear approaches as opposed to a linear approach such as PCA. The most significant disadvantages are in the lack of ease of computation and the absence of a simple forward backward transformation between the low-dimensional representation and the actual data set. In order to use these advantages of linear techniques, extensions to the PCA have been proposed. Among these extensions is the Vector Quantization PCA (VQPCA) [15]. This technique first partitions the data space into disjoint regions by vector quantization and then performs local PCA about each cluster center. The criterion for partitioning the data is in terms of reconstruction error, i.e., a data point is assigned to the cluster that provides a smaller reconstruction error after PCA. This process is iteratively repeated till a convergence in terms of the squared error is obtained. This hard partitioning of data into clusters before dimensionality reduction leads to loss of the global information present in the data. We would like to both exploit the local as well as global information present in the data and so prefer a soft partitioning of the data. The idea of soft partitioning the data while training local PCAs has been examined by Tipping and Bishop [16]. They use an extension of the PCA called the Probabilistic PCA (PPCA). They first approximate the data probability density using a mixture of Gaussians and then for each one of these Gaussian components a PPCA is computed and is used to represent the data. The problem with their approach is the fact that the error due to dimensionality reduction is not explicitly minimized, instead the likelihood of observing the data given the model is maximized. This may lead to poor reconstruction performance of the model.

We propose a linear modeling scheme that is an extension to the PCA called *Mixture of Principal Components (MPC)*. Similar to the way that a Gaussian mixture models the data distribution, the MPC automatically model the data using a mixture of eigenspaces, thereby capturing the variations in the data more efficiently. We compare the performance of the MPC with the PCA as well as with the previously proposed extension to the PCA, the VQPCA. We use the MPC to model regions of interest in video sequences, using faces as regions of interest as illustration and show the improvement in performance of using such a model based scheme over using the conventional error concealment scheme.

In order to illustrate other applications for the MPC, we use the task of face recognition and the task of face tracking as examples. We use the MPC to model face data with lighting and pose variations and use these trained models in a face recognition system. We highlight the improvement of recognition performance using the MPC over the PCA. We also describe a scheme for foreground background segmentation using the MPC to model the background and how this segmentation result may be used to obtain robust tracking performance.

We realize that the MPC may be used to model any kind of data with large amounts of variation thereby making it suitable for any task requiring statistical modeling tools. The use of MPC is also not limited to a particular domain. For instance in this thesis we use the MPC to model pixel data representing objects of interest. However, this modeling scheme may be used without modification to model any other different kinds of data, for instance frequency domain coefficients, wavelet coefficients, etc, thereby making it a very general modeling framework.

1.3. Modeling VBR Video Traffic

Modeling variable bit rate (VBR) video traffic is important as it allows video coders and network designers to estimate the parameters of networks like packet loss probabilities and end-to-end delays. This information may be incorporated as part of the coding process to improve video coding performance. Video traffic has many characteristics that an accurate model needs to capture. Video traffic consists of data from different types of frames, Inter or Intra coded, and from different kinds of scenes with varying activity levels and lengths.

There are different kinds of VBR video traffic as defined by Lakshman et al in [21]. These include unconstrained (U)-VBR traffic, shaped (S)-VBR traffic, constrained (C)-VBR and feedback (F)-VBR depending on whether these are generated using a video encoder that is aware of the network or buffer state and uses this information during video coding. Most of the modeling work has been focused on modeling U-VBR traffic. In all of the following discussion whenever we say VBR traffic, we are referring to U-VBR traffic.

Several models for VBR video traffic have been proposed in literature. Maglaris et al [22] have proposed a model for the coding bit rate of a single video source using interframe predictive coding. Sen et al [23] propose models for different activity levels using correlated Markov models and use queuing analysis to estimate the packet loss and delay. Yegenoglu et al [24] propose a model for VBR video using a time dependent autoregressive (AR) model to represent data from different activity levels. Izquierdo and Reeves [25] have performed a survey of different statistical models proposed to model VBR video traffic.

Most of the work described in the literature does not explicitly take into account the presence of different types of frames. Some work by Doulamis et al [26] models I, P and B frames explicitly with an additional layer corresponding to the activity level of the video scene. This is a good model for video traffic. However, they impose a constraint of a fixed Group of Pictures (GOP) structure, which is very restrictive. Chandra and Reibman [27] model I and P frames explicitly and allow for a variable GOP structure. However, their model requires a large number of parameters and they do not allow for any temporal correlation or different activity levels for I frames. They also do not include B frames in their model.

In this thesis we introduce several models that are flexible enough to capture the characteristics of the video traffic. We describe models with I, P and B frames that are doubly Markov in nature, to account for trace having different activity levels as well as different types of frames. We also use autoregressive (AR) processes to capture the temporal correlations between successive frames. These models are extensions of models for I and P frames that we introduced in [28], [29] and [30]. We examine both the stochastic properties of the trace, e.g., the autocorrelation function, as well as the delay and loss probability encountered by the trace using network simulations. We show that the trace generated by our model can predict the delay and packet loss probability encountered by real data accurately.

We realize that the performance of any model may be significantly improved using more accurate prediction for the probability density function (pdf) of the data. Hence we examine ways of creating AR processes with pdfs corresponding to the actual data pdfs. We relate this problem of creating AR processes with a specified pdf with solving a two-scale dilation equation. A lot of work has been done in two-scale dilation equation theory by Daubechies and Lagarias [31] and by Strang [32], who provide some sufficient conditions for the solution to exist and the ways to find this solution. We highlight how we can use those results to create some simple pdfs. We also show that we can directly use the relationship between the noise pdf and the AR process pdf to create the desired pdf. We then create traces with accurately modeled pdfs and highlight the improvements in modeling performance in terms of the better prediction of network parameters such as delay and loss probability.

1.4. Organization of Thesis

This thesis is organized as follows. Chapter 2 describes the fundamentals of video coding algorithms. These algorithms are used across many of the prevalent video coding standards. We then build a block diagram of the encoding and decoding process.

Chapter 3 describes the optimization of the mode decisions part of the encoding process. We build a classification based framework for making mode decisions that are optimal in terms of minimizing a cost defined in terms of one or more parameters of the complexity-quality-bit rate tradeoff.

Chapter 4 introduces the MPC, a general statistical framework for capturing variations in data efficiently. We use the MPC as a model and show improvements over current concealment schemes.

Chapter 5 includes the description of the modeling of video traffic. We try to capture the stochastic variations in the real traces using doubly Markov models and autoregressive processes. We also highlight the need to capture accurately the probability density of the real data and examine the problem of creating an autoregressive process with a desired probability density.

We finally conclude with a summary of the contributions of the thesis and some future directions for research.

2. Fundamentals of Video Coding

This chapter introduces the framework into which all the work in this thesis may be included. We start by describing the fundamentals of video coding and use these building blocks to build the complete video encoder and decoder. We also briefly introduce the prevalent video coding standards. We then describe the problems we are trying to solve and show specifically which parts of the coding problem we are attempting to solve.

Video coding translates video sequences into an efficient bitstream. This translation involves the removal of redundant information from the video sequence. Video sequences contain two kinds of redundancies, spatial and temporal. Spatial redundancy refers to the correlation present between different parts of a frame. Removal of spatial redundancy thereby involves looking within a frame and is hence referred to as Intra Frame Coding. Temporal redundancies, on the other hand are the redundancies present between frames. At a sufficiently high frame rate it is quite likely that successive frames in the video sequence, are very similar. A lot of information present in a frame is also present in the frame that preceded it. Hence, removal of such temporal redundancy involves looking between frames and is called Inter Frame Coding.

A video frame may be Intra (I), Predictive (P) or Bidirectionally-predictive (B). An I frame is coded in isolation from other frames using transform coding, quantization and entropy coding. A P frame is predictively coded, which means that it is coded using motion compensation followed by transform coding and entropy coding. A B frame is predicted bidirectionally, which means that the prediction is formed using both its previous frame as well as the successive frame. An I frame is often used to efficiently code frames corresponding to scene changes, i.e. frames that are different from preceding frames and therefore cannot be easily predicted. Frames within a scene are similar to preceding frames and hence may be coded predictively as P or B for increased efficiency. Frames between two successive I frames, including the leading I frame, are collectively called a group of pictures (GOP). We illustrate a GOP in Figure 2.1.

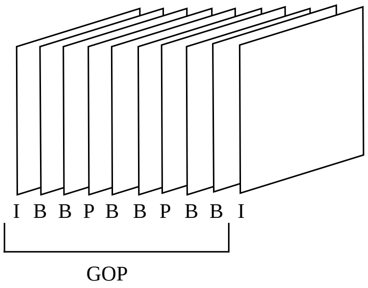

Figure 2.1. Group of Pictures

In Figure 2.1 the group of pictures illustrated has one I frame, two P frames and six B frames. Typically, multiple B frames are inserted between two consecutive P or between I and P frames and we show this in Figure 2.1.

Block based coding is a very popular approach to video encoding. In such approaches the pictures are sub-divided into smaller units or blocks that are processed one by one, both by the decoder and the encoder. These blocks are processed in the scan order as shown in Figure 2.2.

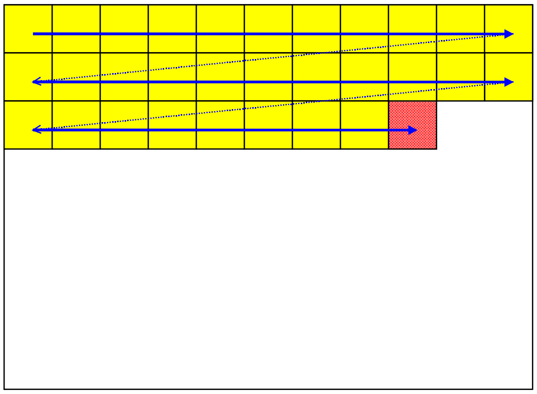

Figure 2.2. Scan order of blocks in a frame

As shown in Figure 2.2, the blocks are processed left to right and top to bottom. Spatial redundancy is removed through the use of Transform Coding techniques. Temporal redundancy is removed through the use of Motion Estimation and Compensation techniques. Specifics of some of the methods to remove spatial and temporal redundancies, with special emphasis on block based coding schemes, are described in the following sections.

2.1. Removal of Spatial Redundancy: Transform Coding

Transform coding has been widely used to remove redundancy between data samples. In transform coding, a set of data samples is first linearly transformed into a set of *transform coefficients*. These coefficients are then quantized and entropy coded. A proper linear transform can de-correlate the input samples, and hence remove the redundancy. Another way to look at this is that a properly chosen transform can concentrate the energy of input samples into a small number of transform coefficients, so that resulting coefficients are easier to encode than the original samples.

The most commonly used transform for video coding is the discrete cosine transform (DCT) [33] and [34]. Both in terms of objective coding gain and subjective quality, DCT performs very well for typical image data. The DCT operation can be expressed in terms of matrix multiplication as $\mathbf{Y} = \mathbf{C}^T\mathbf{X}\mathbf{C}$ where $\mathbf{X}$ represents the original image block, and $\mathbf{Y}$ represents the resulting DCT coefficients. The elements of $\mathbf{C}$, for an 8×8 image block, are defined as

$$C_{mn} = k_n \cos\left[\frac{(2m+1)n\pi}{16}\right] \text{ where } k_n = \begin{cases} 1/(2\sqrt{2}) & \text{when } n = 0 \\ 1/2 & \text{otherwise} \end{cases}$$

After the transform, the DCT coefficients in $\mathbf{Y}$ are quantized. Quantization involves loss of information, and is the operation most responsible for the compression. The quantization step size can be adjusted based on the available bit rate and the coding modes chosen. Except for the DC (zero frequency) coefficients that are uniformly quantized with a fixed step size, a "dead zone" is used while quantizing all other coefficients. The dead zone corresponds to increasing the interval between the two steps around zero and is used to remove noise around zero. The input-output relations for the two cases are shown in Figure 2.3.

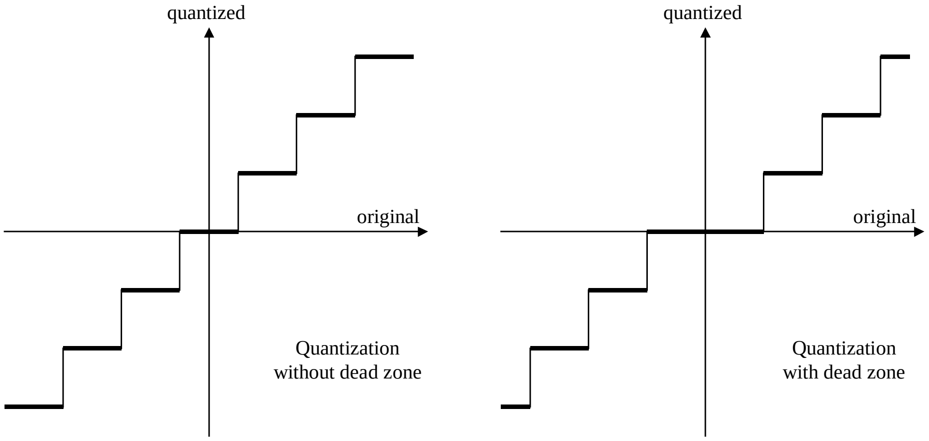

Figure 2.3. Quantization with and without "dead zone"

We can see from Figure 2.3, that quantization is a lossy process with all information between within two steps being represented using one value. This loss of information contributes significantly in the compression process. The dead zone around zero is shown in the I/O relationship on the right. The quantized 8×8 DCT coefficients are then converted into a one-dimensional (1D) array for entropy coding. Figure 2.4 shows the scan order used in many video coding standards for this conversion.

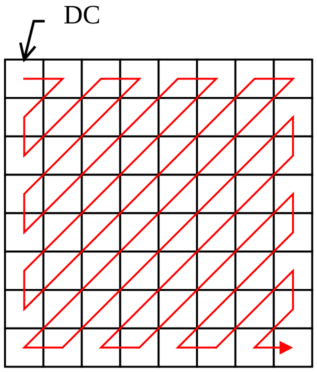

Figure 2.4. Scan order of DCT coefficients

The scan order shown in Figure 2.4 involves scanning the coefficients in increasing order of spatial frequency. This is because most of the energy is concentrated in the low frequency coefficients for typical video sequences with the high frequency coefficients being very small and usually quantized to zero. Therefore, such a scan order can create long runs of zero coefficients, which is important for efficient entropy coding.

Entropy coding takes these quantized coefficients and converts them to bits efficiently. There are different algorithms used for entropy coding. For instance the H.263 video coding standard uses Huffman coding or Arithmetic coding. A brief description of Huffman coding as used in the H.263 standard is described below.

The resulting 1D array is then decomposed into segments, with each segment containing some (this number may be zero) zeros followed by a nonzero coefficient. Let an *event* represent the three values (*run, level, last*). “Run” represents the number of zeros; “level” represents the magnitude of the nonzero coefficient following the zeros and “last” is an indication of whether the current non-zero coefficient is the last non-zero coefficient in the block. A Huffman coding table is built to represent each event by a specific codeword, i.e., a sequence of bits. Such a table is often called a variable length coding (VLC) table. Events that occur more often are represented by shorter codewords, and less frequent events are represented by longer codewords. The aprioiri probabilities for the different events are determined using simulations and examining typical video data transform coefficients. This coding process is sometimes called “run-length coding.” An example of the VLC table is shown in Table 2-1.

Table 2-1. Partial VLC Table for DCT coefficients

LAST	RUN	|LEVEL|	VLC CODE
0	0	1	10s
0	0	2	1111 s
0	0	3	0101 01s
0	0	4	0010 111s
0	0	5	0001 1111
0	0	6	0001 0010
0	0	7	0001 0010
0	0	8	0000 1000
0	0	9	0000 1000
0	0	10	0000 0000
0	0	11	0000 0000

From the table we can see that more frequently occurring symbols, i.e. those with a small magnitude have fewer bits assigned to them, leading to efficient coding.

2.2. Motion Estimation and Compensation

Motion compensation is used to remove the temporal redundancy present in video sequences. When the frame rate is sufficiently high, there is a great amount of similarity between successive frames. Hence, it is more efficient to code the difference between frames, rather than the frames themselves. An estimate for the frame being coded is obtained from the previous frame and the difference between the prediction and the current frame is coded. This concept is similar to Predictive and Differential Coding techniques.

Motion compensation is used to create a prediction for the current frame from the previous frame. The first step is to estimate the motion of objects between frames and then to use this information to build a prediction. The focus of this thesis is on block based coding schemes, so the smallest coding unit for which motion compensation is performed is a block (16×16 region), also sometimes referred to as a *macroblock* in standards.

The motion estimation process involves trying to find a good match for every block in the current frame, in the previous frame. Each block in the current frame is compared to areas of its size in a specified search space in the previous frame, and the best matching area is selected. The process of motion estimation is highlighted in Figure 2.5.

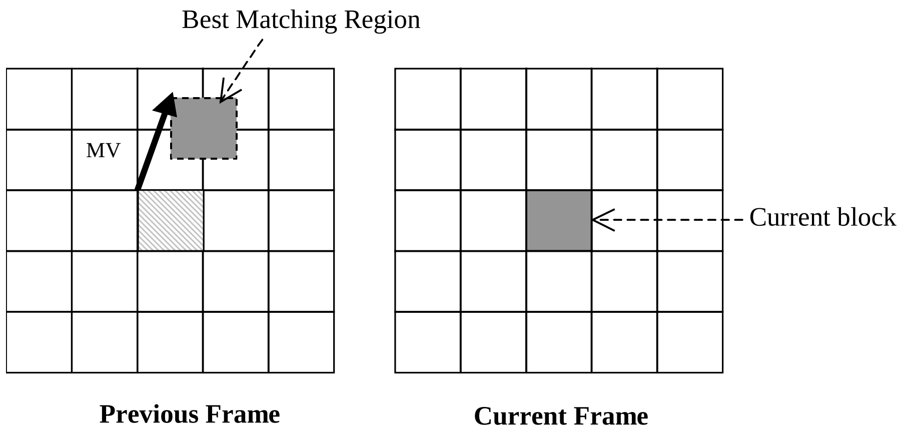

Figure 2.5. Motion Compensation

In Figure 2.5 the dark region on the left is the best matching region for the current block, shown on the right. The lightly shaded region on the left is the location of the current block in the previous frame. The displacement between the position of the current block and the best matching region is called the motion vector (MV) and is also shown in the figure using a bold arrow. This `best matching area is then offset by the MV to obtain a prediction for the current block. In most cases, it is not possible to find an exact match, but the prediction area is usually similar to the block with the residue between the two being small. This residue is computed and coded using the transform coding procedure. More information about motion compensation can be found in [35]and [36]. There is a lot of work in literature on efficient motion estimation strategies since motion estimation is a computationally intensive part of the video encoding. Some of our work on motion estimation may be found in [37].

Sometimes, it is not possible to find a good match for a particular block; this happens especially for blocks in frames that lie across scene changes, or in sequences with large motion. For such blocks the residue itself may be very large and hence the transform coefficients of the residue may be as large or larger than the transform coefficients of the block itself. In such cases it is better to not do motion compensation. The encoder is thus allowed the flexibility to decide for each block, whether it wants to do motion compensation and transform coding of the residue, or just transform coding of the block itself. This decision is called the *Intra-Inter mode decision*. However, in most cases, a saving is accomplished in the bits to code the residue.

Besides the residue we also need to send the information regarding construction of the prediction frame to the decoder, i.e., the motion vectors. The decoder can then use these to reconstruct the prediction frame and add to it the residue (this is transform coded) to obtain the current frame. In order to avoid a large increase in the bitstream size because of these motion vectors, these are also differentially coded.

2.3. Video Codec Block Diagram

All these basic coding algorithms may be put together to form a block diagram representation of the video encoder and decoder. This is shown in Figure 2.6.

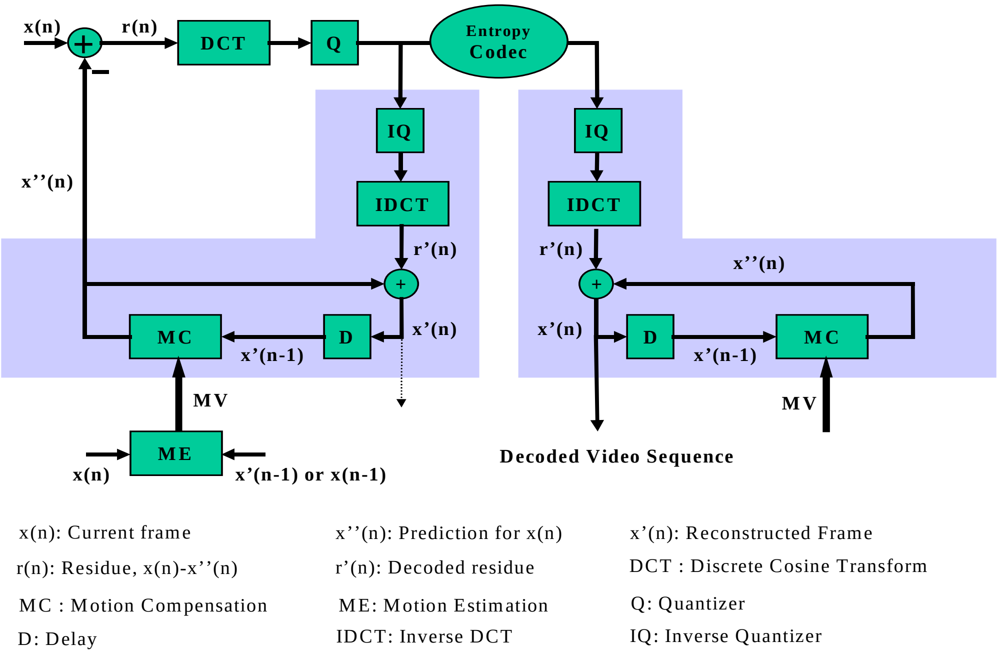

Figure 2.6. Block Diagram of Encoder and Decoder

As may be seen from Figure 2.6, the decoder is contained as part of the encoder, shown as the two L-shaped blocks. This done in order for the encoder to use as reference the same frames as seen by the decoder i.e., instead of using the actual previous frame as reference while coding the current frame, the encoder uses the decoded version of the previous frame in order to be consistent with the decoder. More details on the fundamentals of video coding may be obtained from our book chapter [38].

As mentioned before, the goal of this thesis is to optimize the video coding process in terms of the complexity-quality-bit rate tradeoff. We examine both encoder optimizations as well as decoder post-processing to achieve this goal. We realize that all these optimizations are dependent on feedback from the network regarding its availability and status. Therefore, we focus on modeling the video traffic so that we may use these traffic models as probes to determine this network information.

We redraw the video coding block diagram shown in Figure 2.6, in order to highlight our research areas and show this in Figure 2.7.

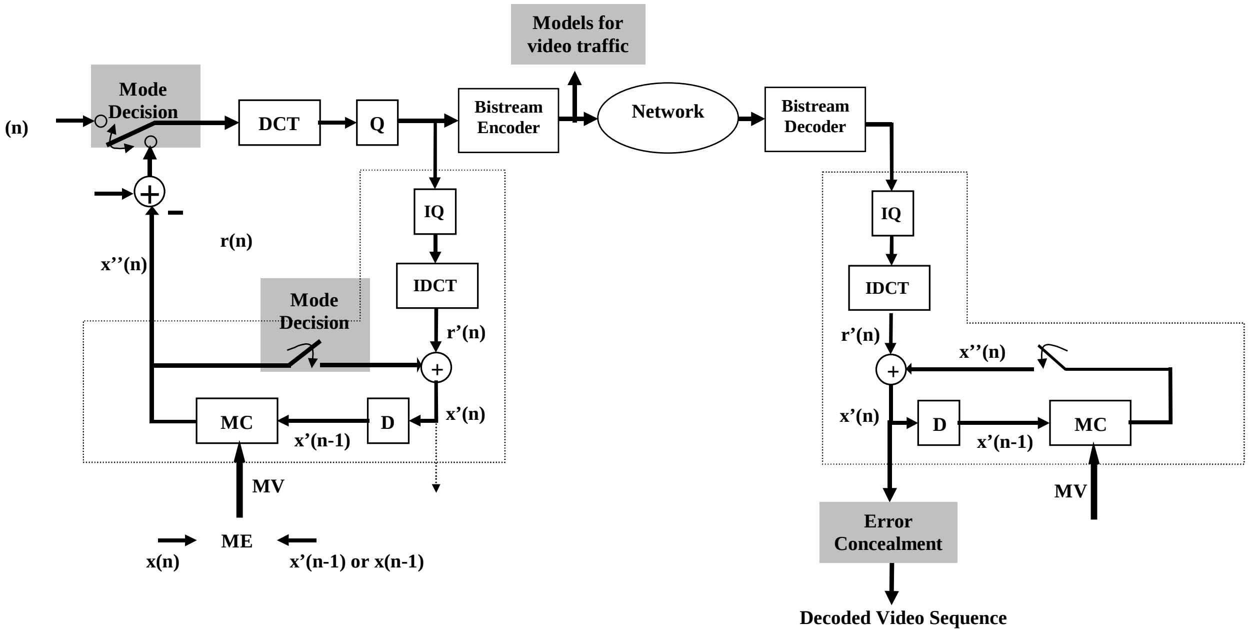

Figure 2.7. Video codec optimization research

In Figure 2.7 we show the video codec block diagram with some additions, shown as the highlighted blocks, Mode Decisions, Error concealment and Models for video traffic. These highlighted blocks correspond to the focus of this thesis. Mode decisions involve selection of coding parameters and modes to meet certain constraints. Error concealment involves post-processing the received video sequence to reduce/remove visible artifacts in the video caused due to errors during transmission over the network. Modeling the video traffic involves capturing the statistical parameters of video traffic as accurately as possible.

2.4. Video Coding Standards

There are two broad sets of video coding standards, the H series standards developed by the International Telecommunication Union (ITU-T) and the MPEG series standards developed by the International Organization for Standardization (ISO).

The ITU-T standards are defined by volunteers in open committees and are agreed upon based on the consensus of all the committee members. H.263 is the latest in the series of low bit rate video coding standards developed by and was adopted in 1996. It combined the features of MPEG and H.261 [39] (an earlier standard developed in 1990) for very low bit rate coding. H.263

version 2 [40] or H.263+ was adopted in early 1998 and is the current prevailing standard from ITU-T. In this thesis whenever we mention H.263, we use it to refer to the H.263 version 2.

The other major organization involved in the development of standards is the ISO. The ISO standards are the MPEG-1, 2 and 4. The most recently released standard is the MPEG-4 [41] that was adopted in 1998. The ISO is currently working on the next standard MPEG-7. Both these organizations have defined different standards for video coding. These different standards are summarized in Table 2-2. The major differences between these standards lie in the operating bit-rates and the applications they are targeted for. Each standard allows for operating at a wide range of bit-rates, hence each can be used for a range of applications. All the standards follow a similar framework in terms of the coding algorithms, however there are differences in the ranges of parameters and some specific coding modes.

Table 2-2. Video Coding Standards

Standards Organization	**Video Coding Standard**	**Typical Range of Bit Rates**	**Typical Applications**
ITU-T	H.261	p×64 kbits/s, p=1…30	ISDN Video Phone
ISO	IS 11172-2 MPEG-1 Video	1.2 Mbits/s	CD-ROM
ISO	IS 13818-2 MPEG-2 Video[1]	4-80 Mbits/s	SDTV, HDTV
ITU-T	H.263	A wide range	PSTN Video Phone
ISO	CD 14496-2 MPEG-4 Video	24-1024 kbits/s	A wide range of applications
ITU-T	H.26L	< 64 kbits/s	A wide range of applications

Standards also include recommendations for the choice of some parameters and algorithms during the video coding process. These are included in Test Models (H Series) or Verification Models (MPEG) provided by the standards committees. Among some of the latest such recommendations are the Test Model Near-Term (TMN-10) [42] of the H.263 standard and the Verification Model (VM-14) [43] of MPEG-4.

[1] ITU-T also actively participated in the development of MPEG-2 Video. In fact, ITU-T H.262 refers to the same standard and uses the same text as IS 13818-2.

3. Classification Based Framework for Mode Decisions

This chapter focuses on optimizing the mode decisions part of the encoding process. We introduce a classification based framework to make these mode decisions to optimize a cost. We use the Intra-Inter mode decisions, and the decision to skip or code a frame to meet a specified target rate, to highlight the improvements obtained by using such a classification based technique.

Inherent in the coding process are many mode decisions that improve one or more parameters of the complexity-quality-bit rate tradeoff. Most mode decisions in the coding process may be viewed as binary hypothesis testing problems where one of two optional modes is selected using a decision criterion. The goal of a mode decision is to minimize a cost that may be defined in terms of one or more of the parameters of the complexity-quality-bit rate tradeoff. Mode decisions may be at any level of the coding process. For instance, some decisions need to be made at the frame level, while some decisions need to be made at the block level. Some examples of these decisions may be the decision to code a block predictively (Inter) or without reference (Intra) or the decision to code or skip a frame given that we have a certain target bit rate to meet.

The optimal mode decision is typically data dependent. In theory, for each mode decision, we can try all the possible modes, evaluate the cost corresponding to each mode, and choose the one with the smallest cost. However, such an exhaustive search approach is impractical due to its complexity. An alternative is to identify features that can be easily computed from the video data, and are good indicators of which mode would be optimal. In order to do so, we first collect video data and use exhaustive search to “label” the data with the optimal mode decisions. We then estimate probability density functions for these features under different hypotheses, corresponding to the different options in the mode decision. We then transform this minimization of cost to a more traditional error probability minimization problem and use standard classification techniques like the likelihood ratio test to solve it. We evaluate the performance of this classification approach using two mode decisions that are part of the video coding process. We first use the Intra-Inter mode decision as an example and then focus on the mode decision for skipping or coding a frame for rate control purposes. We extend our work on rate control to scalable video coding in order to evaluate the performance of our scheme under lossy network conditions. We create an enhancement layer corresponding to the base layer generated using our rate control strategies and examine the reconstructed video quality over different simulated packet loss scenarios. Effectively, this is equivalent to choosing between SNR scalability and temporal scalability adaptively, depending on the video data.

The chapter is organized as follows. Section 3.1 includes the discussion of the general classification based approach to making mode decisions using the Intra-Inter mode decision as an example. Section 3.3 contains the use of this approach towards the rate control problem using an instantaneous mode decision. Section 3.4 includes the description of the mode decision allowing for looking ahead at one future frame to be coded. We then extend this work on rate control to scalable video coding and this is described in Section 3.5. We then conclude with the analysis of our scheme and describe some future work and possible extensions.

3.1. Classification Based Approach to Mode Decision

In this section we show how to convert a mode decision into a binary hypothesis-testing problem. Let c_o represent the cost for making a decision D_0 and c_1 represent the cost for making decision D_1 for a particular coding unit. These decisions D_0 and D_1 may include mode decisions that may be at different levels of the coding process, i.e., at different coding units. For instance these could be at the macroblock level, with one macroblock being the coding unit, where D_0 could be to decide to code the macroblock using intra coding while D_1 could be to code the macroblock using inter coding. These could also be at the frame level, with one frame being the coding unit, where D_0 and D_1 represent whether to code or skip the frame. The costs c_i could include the number of bits needed to code a block or frame given that we have made a particular decision. The cost could also include the distortion introduced in the decoded video as well as the time needed to encode the video according to the decision. The goal of our strategy is to make the decision that minimizes this cost. So every time we need to make a decision we want to make one that results in the smaller cost. This strategy may be summarized as below.

$$\begin{cases} \text{Choose } D_0 \text{ if } c_0 < c_1 \text{ or } \mathrm{c}_0 - c_1 < 0 \\ \text{Choose } D_1 \text{ if } c_1 < c_0 \text{ or } \mathrm{c}_0 - c_1 > 0 \end{cases}$$

3.1.1. Transforming Mode Decision into Classification Problem

In principle, to make the optimal mode decision one can try all the modes and choose the mode that has the lowest cost. However, computing the actual costs c_i before making a decision is very computationally intensive as this involves trying either decision to determine the cost. In order to reduce computational burden for the decision scheme we would like to identify features that provide a good estimate of the cost for a decision, but do not require as much computation to evaluate. We would, thus like to identify features that let us estimate which of the two following hypotheses H_0 or H_1 is true where

$$H_0 : c_0 - c_1 < 0$$
$$H_1 : c_0 - c_1 > 0$$

For each coding unit we identify K features that we group together in a feature vector $\mathbf{x} = [x_0, x_1, \ldots x_{K-1}]^T$. In the optimal scenario we could find features that perfectly represent the cost needed for a decision. However, in most practical applications such features are difficult to find. In most practical applications, the decision strategy thus becomes sub-optimal in terms of minimizing the cost, however we are willing to settle for this sub-optimality as along with this comes the benefit of small computational requirements.

We want to build a classifier that would take as input the feature vector $\mathbf{x}$ of each coding unit and come up with the probability that H_0 or H_1 is true, which would then enable us to make a decision appropriately. We can come up with such a classifier by training it with sample data. Suppose we collect a set of M coding units with corresponding feature vectors $\mathbf{x}_0, \mathbf{x}_1 \ldots \mathbf{x}_{M-1}$ and associated with each of these feature vectors is a cost difference $d_i = c_{0i} - c_{1i}$ with c_{0i} and c_{1i} being the true costs for decisions D_0 and D_1 respectively for the *i-th* coding unit. Let P of these coding units have $d_i < 0$, i.e., corresponding to when H_0 is true and Q of these coding units have $d_i > 0$, corresponding to when H_1 is true, with $P + Q = M$. For purposes of illustration, we show these two sets of training vectors for a *one-dimensional* feature vector in Figure 3.1.

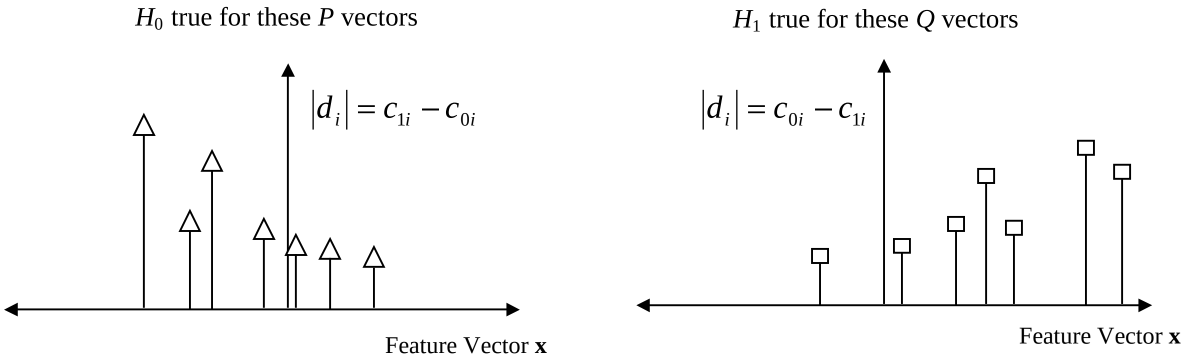

Figure 3.1. Illustration of feature vectors

In Figure 3.1, the x-axis for both the cases corresponds to the value of the feature vector $\mathbf{x}$ (we are using a 1-D vector for illustration) for each coding unit while the y-axis corresponds to the magnitude of the cost difference $|d_i|$ between the two mode decisions for that coding unit. Feature Vectors of coding units for which H_0 is true are shown on the left and are represented with triangles while feature vectors of coding units for which H_1 is true are shown on the right and represented with squares.

The magnitude difference $|d_i|$, for any coding unit, corresponds to the additional cost that we need to pay if we make the wrong mode decision for that unit. For instance if we choose to make a decision D_1, instead of the right decision D_0, for one of the coding units represented by triangles, we incur a cost c_{1i} instead of incurring the smaller cost c_{0i} and so pay an additional cost $|d_i| = c_{1i} - c_{0i}$.

We may put together the two plots from Figure 3.1 to obtain the entire training feature space and this is shown in Figure 3.2.

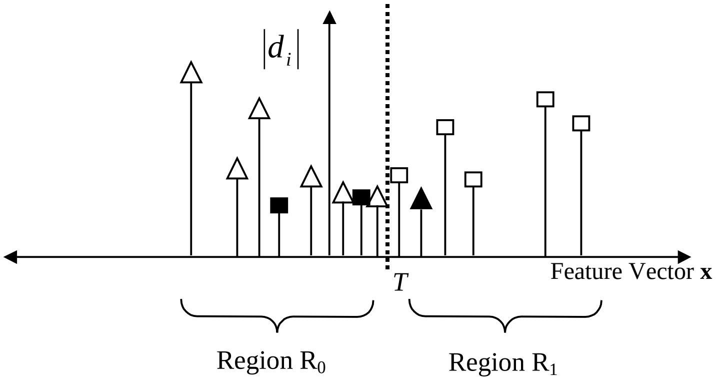

Figure 3.2. Training feature space

All the feature vectors corresponding to the M coding units are shown on the same plot and we use triangles and squares as before to separate the data into the two classes. We now want to partition this feature space using our classifier so that the total additional cost that we have to pay for misclassification, i.e., making the wrong decision for any coding unit, is as small as possible. For instance we may partition this space into two regions, R_0 and R_1, using the threshold *T* as shown in Figure 3.2. We choose to make decision D_0 for all coding units with feature vectors to the left of the threshold, i.e., in R_0, and decision D_1 for all coding units with feature vectors to the right of the threshold, i.e., in R_1. For such a case we pay the additional cost $|d_i|$ for each of the misclassified coding units, i.e., squares in R_0 and triangles in R_1. In Figure 3.2 we show these as dark triangles and squares, as opposed to the lightly colored ones, for which we make the right decision.

We would like our mode decisions to result in as small a total cost as possible. In order to do this we need to choose R_0 and R_1 so that we minimize the total additional cost from making wrong mode decisions. This kind of a problem of partitioning the feature space in order to minimize the total cost is reminiscent of traditional classification theory, except that in traditional classification theory the vertical axis corresponds to the probability densities of the feature vectors under the different hypotheses, while in our problem this corresponds to the additional cost we have to pay when we make a wrong decision. Hence, in order to use traditional classification techniques to solve our problem, we need to modify our problem to fit the classification scenario.

Our problem of minimizing the total additional cost may be mathematically written as in equation (3.1).

$$\min_{R_0,R_1}\left[\sum_{\substack{\mathbf{x}_i\in R_0\\ d_i>0}}|d_i|+\sum_{\substack{\mathbf{x}_i\in R_1\\ d_i<0}}|d_i|\right] \tag{3.1}$$

These two regions, R_0 and R_1 should together span the entire space, and have no common sub-regions. However, we impose no constraints on the shapes of these regions as long as they satisfy this minimization requirement. These regions may consist of non-contiguous sub-regions and the boundaries between them may be arbitrarily shaped. Hence we formulate the problem of minimization in terms of choosing the regions and not in terms of specifying linear boundaries or thresholds separating them.

Let $N_0 = \sum_{d_i<0}|d_i|$ and $N_1 = \sum_{d_i>0}|d_i|$. Our minimization problem of equation (3.1) may be equivalently rewritten as follows.

$$\min_{R_0,R_1}\left[\frac{N_1}{N_0+N_1}\sum_{\substack{\mathbf{x}_i\in R_0\\ d_i>0}}\frac{|d_i|}{N_1}+\frac{N_0}{N_0+N_1}\sum_{\substack{\mathbf{x}_i\in R_1\\ d_i<0}}\frac{|d_i|}{N_0}\right] \tag{3.2}$$

As mentioned before, we would like to use standard classification techniques to solve the problem and identify these regions R_0 and R_1. Such techniques involve estimating the probability density function (pdf) of the feature vector under the different hypotheses and then using these pdfs to identify the best regions. However, we cannot just partition the space by examining the feature vector values, as we need to account for minimizing the total cost of misclassification, i.e., the sum of the total heights associated with misclassified vectors. Essentially, we need to convert these feature vectors with the associated heights to vectors that do not have these additional heights, but we need to do this without losing the important information that the heights carry. We may do this transformation by replacing a vector with height $|d_i|$ with $|d_i|$ vectors at that location. In general it is not necessary that all the heights $|d_i|$ are integers, however without loss of generality we can scale them appropriately to make them integers. We can thus modify our feature space and our modified feature space would look as shown in Figure 3.3.

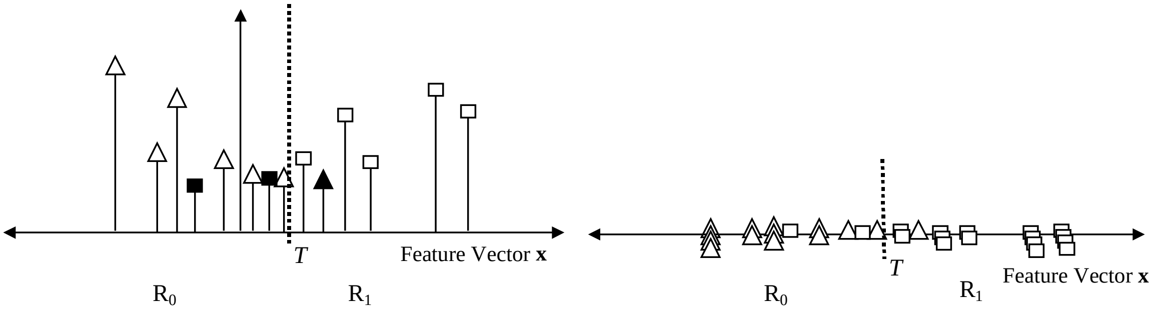

Figure 3.3. Transformation of feature space from old feature space (left) to new feature space (right)

From Figure 3.3 we can see that each vector in the old feature space is replaced by multiple vectors at that location, the number of new vectors being equal to the height associated with the original vector. Now, standard classification techniques may be applied in this new feature space to estimate the pdfs for this new set of vectors. We rewrite our minimization problem in this new feature space and then map it to the well-understood minimum probability of error classification problem. We can thus use results from literature to find the regions that minimize our criterion in this new feature space. Because of the way we transform the old feature space to the new feature space, the regions we find in this new feature space are identical to the regions we desire in the original feature space. This is because none of the training data points are displaced from their original positions. The details of this proposed scheme are presented in the following paragraphs.

In this new feature space N_0 is simply the total number of triangles and N_1 is the total number of squares, where N_0 and N_1 are as defined before. We may define two new hypotheses as following.

$$\begin{aligned} &H_0' : \mathbf{x}_i \text{ belongs to the triangle class} \\ &H_1' : \mathbf{x}_i \text{ belongs to the square class} \end{aligned}$$

Hence, in this new feature space $\frac{N_0}{N_0 + N_1} = P(H_0')$, the probability of a feature vector being a triangle and similarly $\frac{N_1}{N_0 + N_1} = P(H_1')$.

Also $\frac{|d_i|}{N_0} = P(\mathbf{x} = \mathbf{x}_i \mid H_0')$ when $d_i < 0$, because $|d_i|$ is the number of triangles at $\mathbf{x}_i$ and N_0 is the total number of triangles. Similarly, $\frac{|d_i|}{N_1} = P(\mathbf{x} = \mathbf{x}_i \mid H_1')$ when $d_i > 0$. Hence our minimization problem in equation (3.2) may be rewritten in this new domain as follows

$$\min_{R_0, R_1} \left[P(H_1') \sum_{\mathbf{x}_i \in R_0} P(\mathbf{x} = \mathbf{x}_i \mid H_1') + P(H_0') \sum_{\mathbf{x}_i \in R_1} P(\mathbf{x} = \mathbf{x}_i \mid H_0') \right] \tag{3.3}$$

In practice, instead of using these discrete probabilities, we model the data using a continuous pdf consisting of a mixture of Gaussians. This is because in order to classify a feature vector we need to have the probability of occurrence of that vector. However we do not have these probabilities for new input vectors that are not present in the training data set. By modeling the feature vector pdf using a mixture of Gaussians we ensure that any feature vector may be classified. These Gaussian mixtures are trained on the modified feature vectors using the Expectation Maximization (EM) algorithm, more details on which may be obtained from [44]. An example of trained Gaussian mixtures in the modified feature space is shown in Figure 3.4.

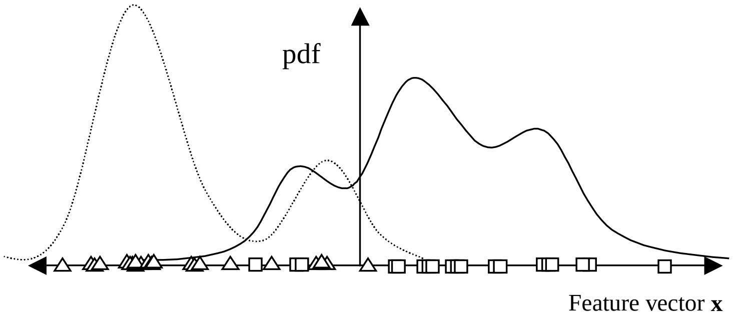

Figure 3.4. Modeling the data by a mixture of Gaussians

The density function drawn with the dotted line corresponds to the data from the triangle class while the density function drawn with the solid line corresponds to the data from the square class. Using our continuous pdfs our minimization problem may be written as in equation (3.4).

$$\min_{R_0, R_1} \left[P(H_1') \int_{\mathbf{x} \in R_0} p(\mathbf{x} \mid H_1')\, \mathbf{dx} + P(H_0') \int_{\mathbf{x} \in R_1} p(\mathbf{x} \mid H_0')\, \mathbf{dx} \right] \tag{3.4}$$

The functions $p(\cdot)$ correspond to the continuous pdf comprising of a mixture of Gaussians. This kind of a minimization problem is equivalent to a minimum probability of error classification scheme in this new feature space and it is well known from literature [9] that the regions may be determined using the likelihood ratio test. The likelihood ratio test may be written as in equation (3.5).

$$\frac{p(\mathbf{x} \mid H_1')}{p(\mathbf{x} \mid H_0')} \underset{H_0}{\overset{H_1}{\gtrless}} \frac{P(H_0')}{P(H_1')} \tag{3.5}$$

Hence, in order to classify a feature vector $\mathbf{x}$, we evaluate the likelihood ratio for it and if this exceeds the threshold obtained from training, we believe H_1 is true and make decision D_1 otherwise we believe H_0 is true and make decision D_0. In summary, this likelihood ration test defines the decision regions in this new space. As we mentioned before, the regions that we determine in this new feature space are identically the regions that we desire in our original feature space. So by transforming our feature space and mapping the problem to a well-understood minimization problem we can obtain the solution we desire.

The entire classification scheme may be summarized as follows. Given the training data and the cost differences, we first transform the feature space to the new feature space and then estimate the apriori probabilities, $P(H_0')$ and $P(H_1')$, as well as the class conditional probability density functions, $p(\mathbf{x} \mid H_0')$ and $p(\mathbf{x} \mid H_1')$, using the EM algorithm to train the Gaussian mixture. Once we have these pdfs, we use the likelihood ratio test for a new input feature vector corresponding to a coding unit and determine which of the two hypotheses is more likely to be true and using this result we make decision D_0 or D_1 for that coding unit.

3.2. Intra versus Inter Mode Decision

This decision is made for every macroblock (a 16×16 region in a frame) in the video sequence. Intra coding involves coding using transform coding followed by quantization and entropy coding. As opposed to this, Inter coding involves building a prediction for the current macroblock using data from the previous frame using motion estimation and compensation and coding the residue using transform coding, quantization and entropy coding. For most macroblocks, Inter coding is more efficient in terms of compression, however this is not always true. When there is a scene change or when we have a high motion sequence the prediction for the macroblock from the previous frame is likely to be poor and in such cases it may be more efficient to use Intra coding as opposed to Inter coding. Hence the encoder has to decide between these for every macroblock. The decision that requires fewer bits is preferred.

As converting to bits is computationally expensive, encoders use features as estimates for the bits. Typically, a measure of the energy (with DC value removed) in the block is used as an estimate of the bits needed for intra coding, while the mean absolute difference (MAD) is used as an estimate for the bits needed for inter coding. The MAD between two 16×16 regions may be defined as $MAD=\frac{1}{256}\sum_{i=1}^{16}\sum_{j=1}^{16}\left|x_{ij}-y_{ij}\right|$. The block *x* represents the current block while the region *y* is the motion compensated prediction for the block from the previous frame. The MAD is thus a measure of the motion compensation performance, with a smaller MAD corresponding to a better motion compensation. Hence the MAD is representative of the number of bits needed to inter code the block.

The mode decision as recommended by the TMN-10 of the H.263 standard is

$$\begin{cases} \textit{Intra Coding if } MAD-E_x>T \\ \textit{Inter Coding otherwise} \end{cases}$$

For a 16×16 block $E_x=\frac{1}{256}\sum_{i=1}^{16}\sum_{j=1}^{16}\left|x_{ij}-m_x\right|$ is used as the measure of energy, m_x is the mean or the DC value of the block and *T* is an empirically found threshold, specified in the TMN as 500.

We also test another feature, the mean removed MAD (mrMAD) as a feature to estimate the bits needed for inter coding. We introduce this feature in [37]. This is very similar to the MAD, except that instead of taking the absolute pixel difference and summing them up we first remove the means from the blocks and then sum up the absolute pixel difference. The mrMAD between two 16×16 regions may be defined as $mrMAD=\frac{1}{256}\sum_{i=1}^{16}\sum_{j=1}^{16}\left|\left(x_{ij}-m_x\right)-\left(y_{ij}-m_y\right)\right|$.

As mentioned before, the block *x* represents the current block while the region *y* is the motion compensated prediction for the block from the previous frame.

In order to collect training data we perform the exhaustive test for the Foreman, Coastguard and Silent video sequences. Snapshots from these sequences are shown in Figure 3.5.

Figure 3.5. Snapshots from Foreman (left) Coastguard (center) and Silent (right)

The sequences are in 176×144 (QCIF) format at a frame rate of 30 Hz. We choose these sequences as possess a variety of motion. Foreman consists of large and uncorrelated motion, especially due to the out of plane motion of the person in the foreground. Coastguard has moderate to large motion, but it is very correlated as it is created by a panning camera. Silent contains low to moderate degree of motion. These sequences are representative of the different kinds of motion that we may encounter with video sequences.

We generate a sequence of values for the features and a sequence of the optimal decisions. These exhaustive tests involve actually computing bits needed for Intra and Inter coding and making the right decision, i.e. the decision resulting in fewer bits. After collecting the sequence of right decisions and the values of the features we correlate these with the decision sequence. From experiments we see that the energy is very well correlated with the bits needed for intra coding (correlation coefficient of 0.87). The MAD and the mrMAD are features representative of the bits needed for Inter coding. They have correlation coefficients of 0.90 and 0.94 respectively with bits for inter coding, independent of sequence. In order to test the suitability of using these features, we correlate these with the optimal decision sequence (one determined using the exhaustive test). The decision sequence is viewed as a sequence of +1s and −1s with +1 corresponding to Intra and −1 corresponding to Inter. More details on our computation of correlation coefficients are included in Appendix A in Section 3.7. The correlation coefficients for each feature are presented in Table 3-1.

Table 3-1. Correlation coefficients of features with decision sequence

Feature	Correlation coefficient with decision sequence
Energy (DC removed)	0.3221
MAD	0.4741
mrMAD	0.5111

From the table we see that the mrMAD is better correlated with the decision sequence than the MAD, hence we choose this as a feature. Although the MAD has a higher correlation with the decision sequence than the energy, it is highly correlated with the mrMAD. As the energy is representative of the bits needed for intra coding, we use this feature, instead of the MAD, along with the mrMAD for our classification scheme.

We collected 79200 feature vectors from these three video sequences of which we used 5000 to train our classifier pdfs and the remaining to test the performance of the scheme. The number of training vectors is small as compared to the test set, but using a larger training set does not improve the performance of our classifier significantly. Using the training data, we train the Gaussian mixtures using to obtain decision regions as shown in Figure 3.6.

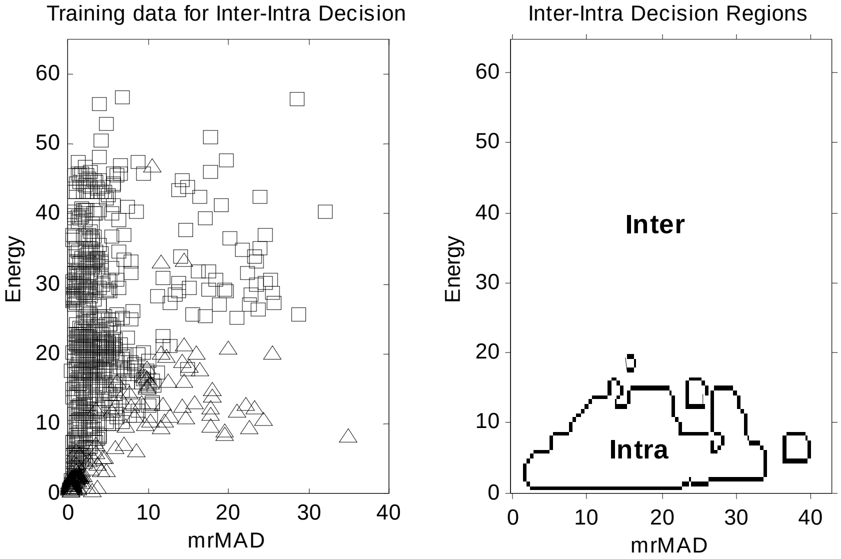

Figure 3.6. Intra-Inter training data (left) and decision regions (right)

The plot in Figure 3.6 shows 1000 training feature vectors on the left with triangles corresponding to vectors that belong to the Intra class and squares corresponding to Inter class. On the right we also show the decision regions that we obtain after training Gaussian mixtures on this training data, with the Intra decision region enclosed inside the black boundary. We can see that the decision regions are representative of the training data and may consist of disjoint sub-regions, as we have for the Intra case. From the training data we can see that a linear decision boundary is not suitable in this case. Thus, imposing no constraint on the shape of the decision boundary aids our classifier in finding a better decision boundary. We achieved 98.2% correct classification using our minimum probability of error classifier. The classification result shows a variation of less than 1% across these different sequences. The classification scheme proposed in the TMN achieves around 92% correct classification, because it uses a linear decision boundary. Hence we can improve the mode decision using this scheme. We then implemented this mode decision in the H.263 framework and observed that the corresponding savings in total bit rate over the TMN decision (including residue bits, motion vector bits and overhead bits) were around 4.5~4.8% for the Foreman, Coastguard and Silent video sequences. In practice, we may train the classifier with such a set of video sequences that are representative of the different kinds of motion and texture that we encounter in real video sequences and the trained classifier may then be used to make the mode decision for any arbitrary test sequence.

3.3. Mode Decision for Skipping or Coding Frames: Instantaneous case

The goal of this mode decision is to decide between skipping and coding a frame in order to maximize the perceived quality while achieving a target bit rate. In order to collect training data for our classifier, we first implement an exhaustive search based mode decision. We compute the effect of skipping as well as coding a frame on the quality *q* and the bit rate *r* before making a decision. In order to measure the perceived quality we use a spatio-temporal quality metric introduced by Wolf and Pinson [45]. We include a description of this metric in the Appendix in Section 3.7. In order to control the rate of coded frames, we change the quantization step size using the inverse quadratic model to relate the bit rate with the quantization step size. This model was proposed in [7] and it relates the rate (r) to the quantization step size (*Q*) using $r = \frac{a}{Q} + \frac{b}{Q^2}$, where *a* and *b* are constants that may be estimated using training data. We found that such a simple model cannot capture the variation of bit rate with quantization step size across many different scenarios, so we train separate models, i.e., parameters *a* and *b*, for low motion, medium motion and high motion sequences. More details about the quality metric can be found in Appendix B in Section 3.7.

Our cost is defined as a combination of the quality and rate, defined as $q + \lambda r$, where the factor λ is adjusted depending on application and we discuss this in more detail later. We compare this cost for the skipping or coding and choose the one that requires the smaller cost. Simultaneously we collect the cost difference and also evaluate some features that we use for training our classifier. Using the method described in Section II, we train the density functions for our features and implement our classification based scheme.

We first describe the exhaustive search based mode decision, after which we describe the features that we choose for the classifier and finally we include the results of our implementation.

In all of the following discussion, the video sequence is represented by a sequence of frames $\ldots X(n-1), X(n), X(n+1)\ldots$ with n representing the time index. Since we are using lossy compression techniques the sequence of decoded frames may be represented as $\ldots \hat{X}(n-1), \hat{X}(n), \hat{X}(n+1)\ldots$. These may not be identical to the original video sequence. The previous decoded frame is used as reference to code the current frame and when a frame is skipped it is replaced by the previous decoded frame.

The steps for the exhaustive search based mode decision are as follows. We first compute the rate and quality when we skip a frame. We replicate the previous decoded frame to simulate skipping the current frame. We then estimate the quality q_1 of this sequence of two frames $\left\{\hat{X}(n-1), \hat{X}(n) = \hat{X}(n-1)\right\}$. The bit rate r_1 is estimated by averaging the bits needed to code the past ten frames, setting the bits for the current frame to zero and multiplying by the frame rate. We estimate the bit rate using a ten frame window as this smooths out the fluctuation due to large or small number of bits to code the current frame.

We then estimate quality and bit rate for coding the frame. We determine the bits available to code the current frame using history information and the target rate. We then estimate the quantization step size needed to code this frame using the inverse quadratic model. Using the previous decoded frame as reference and the computed quantization step size, we code the current frame and reconstruct it. We compute the quality q_2 of this two frames sequence $\left\{\hat{X}(n-1), \hat{X}(n)\right\}$ and the bit rate r_2 as before.

We then compare $q_1 + \lambda r_1$ with $q_2 + \lambda r_2$ and decide which of the two is better. The factor λ can be specified by the user in terms of the relative importance of either the rate or the quality. In our tests we place a greater emphasis on the quality of the sequence. This is done by adjusting λ so that λ times the target rate is 0.5 that is comparable to the range of the quality [-1, 0]. This is acceptable as we already use the quadratic model to try to control the bit rate. This decision strategy may be represented pictorially as in Figure 3.7.

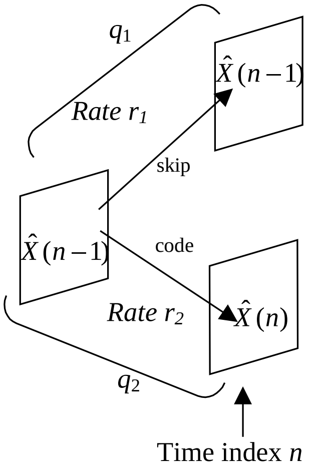

Figure 3.7. Pictorial representation of Exhaustive scheme

This exhaustive scheme is very similar to the Viterbi decoding scheme [46] with no look-ahead allowed, as at every instant in time *n*, costs for both the paths (skipping the frame or coding it) are compared and the best path is chosen, with the other being discarded. In fact this rate control strategy can be extended to allow for look ahead and this is described in Section 3.4. There we also include a greater discussion relating our scheme to the Viterbi decoding scheme.

During this exhaustive mode decision we also evaluate some features. We start with a large set of features and look at the correlation of these features with the decision sequence of the exhaustive search based mode decision to identify the features we use for our classifier. Some of the initial features that we identified are described in the following paragraph.

1) Size of motion vectors. This is computed as the sum of the square length of all the motion vectors in the frame.
2) MAD or the sum of absolute difference (SAD) as a measure of quality of motion compensation. The SAD is a scaled version of the MAD, where we omit the normalization by 256 while summing up the difference. We use the sum of the SAD across all the blocks of the frame.
3) Measure of high frequency energy (HFE) in frame. This is obtained by taking a frame, down-sampling it by a factor 2 horizontally and vertically, then up-sampling it back to the original size and finding the energy in the difference between this and the original frame. This process may be viewed as in Figure 3.8.

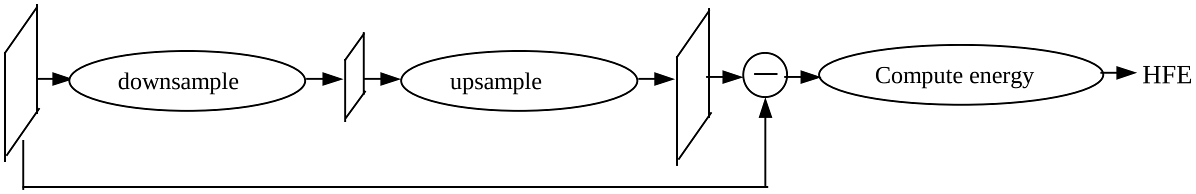

Figure 3.8. Computation of HFE

In the above figure down-sampling includes a pre-processing by a low pass filter and up-sampling includes a post-processing with a low pass filter.

4) Bits available to code current frame. This may be obtained from rate history.

5) Quantization step size used for current frame.

6) Energy in the frame difference between the current frame and previous frame.

We collected these features across the Foreman, Coastguard and Silent sequences, across different target rates using our exhaustive search based mode decision. We then correlate these features with the decision sequence that is viewed as a sequence of +1s and −1s with +1 corresponding to skipping a frame and −1 corresponding to coding it. More details of our computation of these correlation coefficients are included in Appendix A in Section 3.7. The correlation coefficient for each of the features is included in Table 3-2.

Table 3-2. Correlation coefficients for features with decision sequence

Sequence	Target (kbps)	Size of MV	SAD	HFE	Available Bits	Quant. Step Size	Frame difference
Foreman	150	–0.5281	0.0643	–0.4741	0.0324	–0.1551	–0.4612
	300	–0.3342	–0.0200	–0.2761	0.0113	0.0412	–0.2129
	450	–0.4441	–0.0810	–0.3821	–0.251	0.1203	–0.1660
	600	–0.4765	–0.2060	–0.4109	–0.1185	0.355	–0.3132
Coastguard	150	–0.3318	–0.0578	–0.5202	–0.0993	0.2612	–0.0745
	300	–0.4001	–0.0339	–0.3942	–0.4613	0.5244	–0.0299
	450	–0.4114	–0.1320	–0.431	–0.4112	0.4197	–0.0299
	600	–0.4759	–0.1203	–0.4527	–0.4176	0.4003	–0.0868
Silent	150	–0.3914	–0.0218	–0.3452	–0.2147	0.2881	–0.057
	300	–0.3873	–0.1008	–0.3704	–0.2321	0.3092	–0.213
	450	–0.3901	–0.2137	–0.3883	–0.3001	0.3111	–0.2247
	600	–0.3874	–0.0789	–0.3872	–0.1492	0.2427	–0.2019

As can be seen from Table 3-2, most features are negatively correlated with the decision sequence while the quantization step size is positively correlated. This is as expected because small motion vectors, small SAD, small HFE and a small frame difference all imply that the current frame can be very well predicted by the previous frame, thereby meaning that they are similar. This means that we can skip the frame, as we would then replace it with the previous frame. This biases the decision towards not coding the frame or a +1. So a smaller value of each of these features corresponds to a large value in the decision sequence, hence a negative correlation coefficient. A small number of available bits means that the quality of coding the frame will be poor, hence this also tends to bias the decision towards skipping the frame, thereby leading to a negative correlation. On the other hand a small quantization step size indicates that the quality of coding the frame will be good, thereby biasing the decision towards coding the frame and hence leading to a positive correlation coefficient.

Among these features we can see that the size of motion vectors and the HFE have the largest correlation coefficient values across most sequences and most rates. They are also relatively uncorrelated with a correlation coefficient 0.43. Hence, we choose these as representative features for our test. The motion vectors are expensive to compute but we can replace them with motion vectors from the previous frame, as the correlation between them is quite large at 0.87. The HFE for any frame of the sequence needs to be evaluated only once as this can be stored and looked up every time we code the sequence irrespective of the target rates.

We train the density functions, as described in Section 3.1, for these selected features and build our classification based mode decisions. We evaluate these mode decisions over three different sequences, Foreman, Coastguard and Silent. All these sequences are CIF at a frame rate 30 Hz. Of these sequences, Foreman is a high motion sequence, Coastguard is a medium motion sequence and Silent is a low motion sequence.

The results are evaluated using rate distortion curves. Four different target rates (150, 300, 450 and 600 kbps) were chosen for the test. The distortion is measured using the spatio-temporal quality metric that we described earlier. We also compare the rate control using these mode decisions with one that we call No Skip rate control. This scheme tries to control the rate by only changing the quantization step size and skips a frame only when there are no bits available to code it. These results are shown in the following figures.

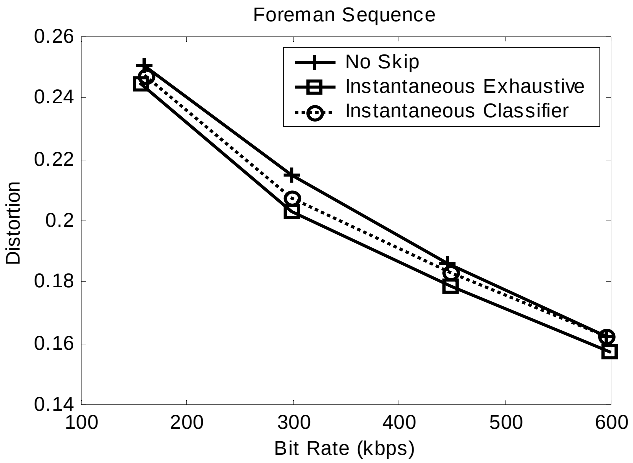

Figure 3.9. Rate Control results for Foreman sequence

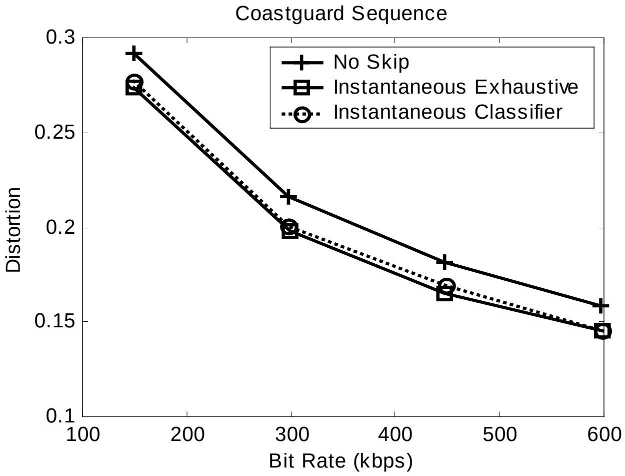

Figure 3.10. Rate control results for Coastguard sequence

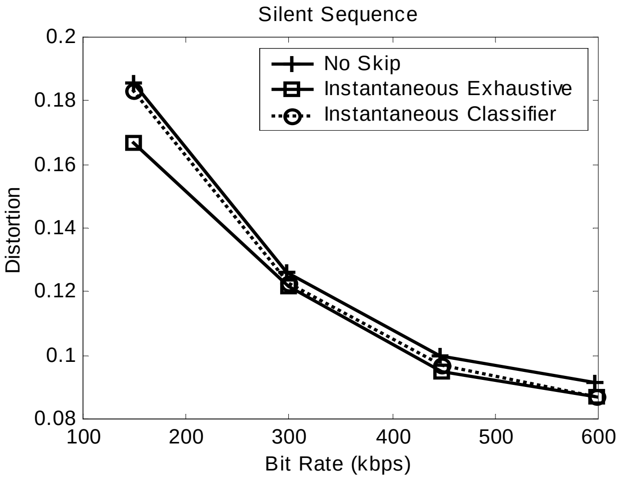

Figure 3.11. Rate Control results for Silent Sequence

We can see from these curves that rate control using both the exhaustive as well as the classification based mode decision perform better than the No Skip rate control as they provide smaller distortion at the same target rate. The distortion is measured using the spatio-temporal quality metric that we described earlier. The performance of the classification based mode decision is close to the exhaustive search based decision. The error probability for the classifier for Foreman is 0.171, for Coastguard is 0.127 and for Silent is 0.131. This leads to the classifier curve for Foreman not being as close to the exhaustive mode decision curve as it is for the other sequences. We can also see that the average percentage improvement in quality across all the rates over the No Skip rate control is smallest for the Foreman sequence. This is because Foreman is a high motion sequence, hence skipping a frame is worse in quality than coding a frame a majority of the time, thereby leading to fewer frames being skipped. In terms of computation requirements the exhaustive mode decision uses roughly 3.5 times the computation as the classification based mode decision. The encoder with the classification based mode decision has a computation complexity within 5% of an encoder with no rate control strategy, where the computation complexity is measured in terms of the time needed for processing the video data.

3.4. Mode Decision for Skipping or Coding Frames: Look-Ahead Case

We extend the mode decision in the previous section by allowing a one-step look-ahead before making a decision. As in the previous section we first describe the exhaustive approach followed by a description of the features that we select and the results for our experiments.

The steps in the exhaustive approach using look-ahead are as follows. We first skip the current frame and replicate the previous decoded frame. The quality and the rate for this set of two frames $\left\{\hat{X}(n-1), \hat{X}(n) = \hat{X}(n-1)\right\}$ are computed as described before. Using this current reconstructed frame as reference, the future frame is both coded and skipped. Quality and rate for the two frame sequences $\left\{\hat{X}(n) = \hat{X}(n-1), \hat{X}(n+1) = \hat{X}(n-1)\right\}$, when we skip the future frame as well and $\left\{\hat{X}(n) = \hat{X}(n-1), \hat{X}_1(n+1)\right\}$, when we code the future frame using this skipped frame as reference, are computed.

We then code the current frame and quality and rate for $\left\{\hat{X}(n-1), \hat{X}(n)\right\}$ are computed. Then using this coded frame as reference the next frame is both skipped and coded. Quality and rate for $\left\{\hat{X}(n), \hat{X}(n+1) = \hat{X}(n)\right\}$, when we skip the future frame and $\left\{\hat{X}(n), \hat{X}(n+1)\right\}$, when we code the future frame, are also computed. This process may be represented pictorially as in Figure 3.12.

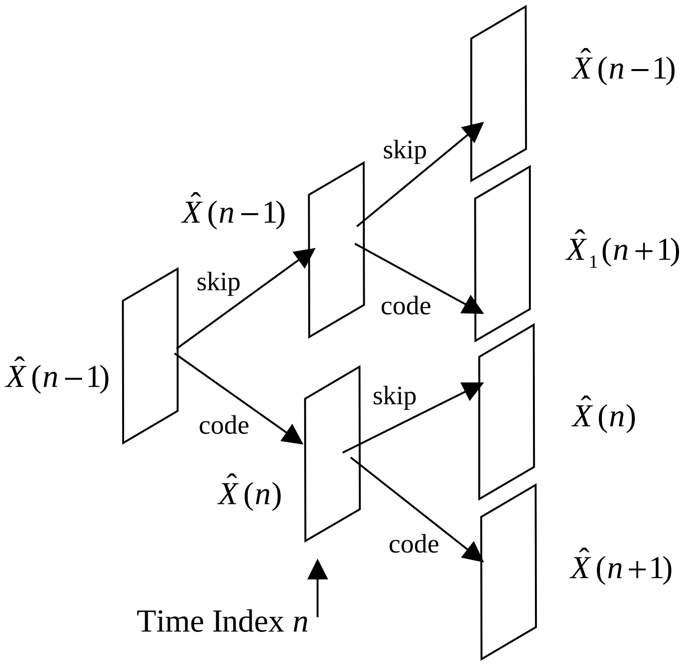

Figure 3.12. All frames generated for look-ahead exhaustive decision

The decision on coding or skipping the current frame is made after looking at the total cost ($q_i + \lambda r_i$) for each of the four paths (skip, code), (skip, skip), (code, skip) and (code, code). The path that provides the best cost is identified and the decision for the current frame is made appropriately. This strategy is very similar to the Viterbi decoding scheme with a one step look-ahead. The similarity is shown in Figure 3.13.

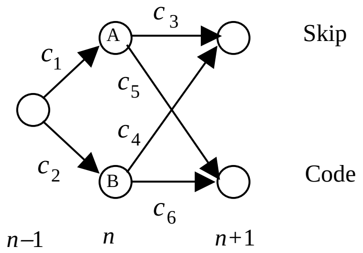

Figure 3.13. Similarity with Viterbi Decoding Scheme

We look ahead one step while trying to make a decision at the current time instant *n*, i.e. decide between nodes A and B. The costs c_i in the figure corresponding to $q_i + \lambda r_i$ are compared i.e. $c_1 + c_3$, $c_1 + c_5$, $c_2 + c_4$ and $c_2 + c_6$ are compared to get the best cost. The node through which this path passes is chosen as the decision for the current frame while the other node is discarded.

For the look-ahead classifier, the feature set that we start with is the same as in the previous section. Of these features we find that the size of motion vectors, the HFE and the quantization step size are the most representative features, i.e. they have the largest correlation coefficients with the decision sequence and have relatively low correlation between themselves. Of these features, identifying motion vectors requires a large amount of computation, so we can replace the motion vector size with the motion vectors from the previous frame. However, it is not good to approximate the future frame motion vectors with those from the previous frame as the correlation between the motion vectors that are two frames apart is not so large. Hence we decide against using motion vector size as a feature for this decision strategy. As against this, the quantization step size for the current frame and those for future frames can be estimated using the model we have, so we prefer to use this. We choose four features for our classifier,

a) Quantization step size for current frame.

b) Estimate of quantization step size for future frame, if we skip the current frame.

c) Estimate of quantization step size for future frame, if we code the current frame.

d) HFE for the current frame.

All these features are easy to compute and as stated before, the HFE for any frame needs to be evaluated only once for any sequence. These look-ahead mode decisions are applied to the Foreman, Coastguard and Silent sequence and the results are compared with the No Skip rate control. These results are shown in the following figures.

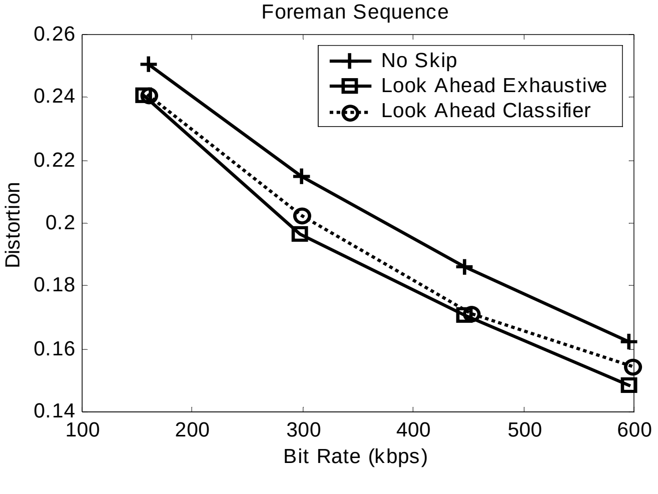

Figure 3.14. Rate control results for Foreman sequence

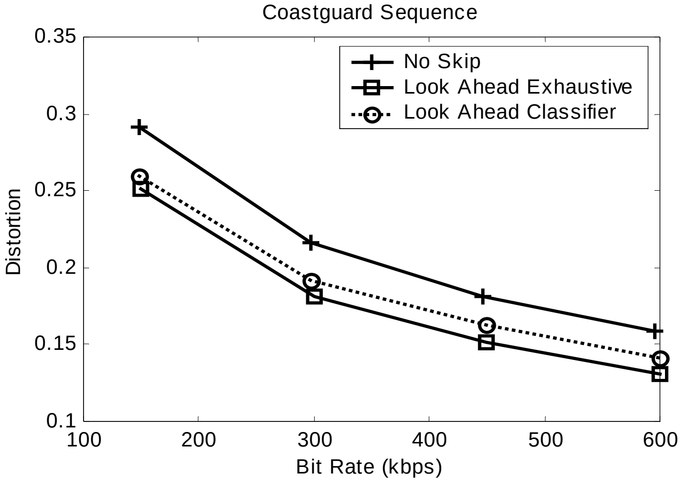

Figure 3.15. Rate control results for Coastguard sequence

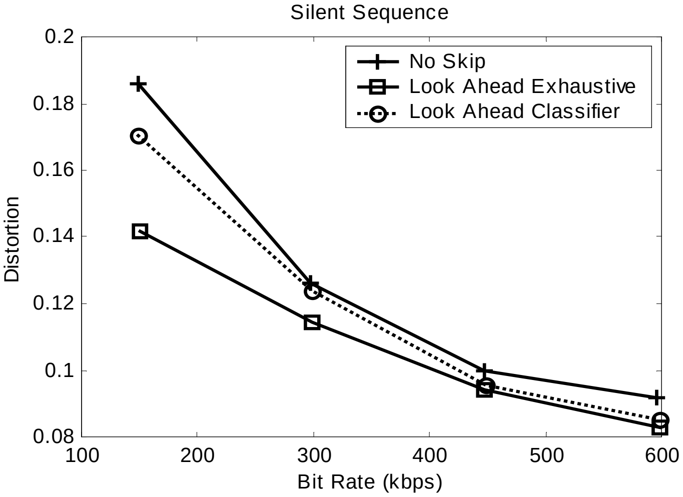

Figure 3.16. Rate control results for Silent sequence

As before, we can see that rate control using both the exhaustive as well as the classification based mode decision perform better than the No Look rate control. The performance of the look-ahead mode decisions is better than the instantaneous mode decisions. The error probability for the classifier for Foreman is 0.168, for Coastguard is 0.202 and for Silent is 0.206 and hence the performance of the classifier is not as good as the exhaustive mode decision. As pointed out earlier the average percentage improvement in quality across all the rates over the No Skip rate control is smallest for the Foreman sequence as it is a high motion sequence. In terms of computation requirements the look-ahead exhaustive mode decision uses roughly 8x times the computation as the classification based mode decision. As before, the encoder with the classification based mode decision has a computation complexity within 5% of an encoder with no rate control. The number of steps that we are allowed to look-ahead can be increased for a greater improvement in performance, as the classification based mode decision does not require a significant increase the computation requirements.

Through our experiments we have shown that we can use classification based strategies to intelligently decide between skipping a frame and coding it to achieve a better rate control strategy than just changing the quantization step size to control rate and use frame skipping only when we have no bits available to code the current frame. All of this discussion was for rate control at the frame level, so we use one quantization step size for the entire frame, however our classification based schemes can easily be extended to block layer rate control, when we can decide to skip or code a block intelligently. Another approach to the frame skipping is to keep the frame rate uniform, but changing it appropriately to meet the target rate. This approach to rate control may also be implemented using our classification based strategy. We may use a classifier to make the decision to change the frame rate after a certain number of coded frames, using the strategy described above. The training data may be collected by exhaustively examining different features under different frame rates and the overall rate-distortion performance.

3.5. Extension to Scalable Coding

Scalable bitstreams are used by video coding schemes to improve the error resilience over lossy networks. The bitstream is partitioned into multiple layers and consists of a base layer and one or more enhancement layers. The base layer is usually assigned the highest priority and error protection and possesses enough information for the decoder to reconstruct the video sequence at a lower resolution, frame rate or quality. The enhancement layers consist of residue information between the base layer and the actual video sequence thereby allowing for reconstruction of the video at a higher resolution, frame rate or quality. There are three different scalabilities supported in the H.263 and MPEG-2 standards, which are the Spatial, Temporal and SNR scalabilities. In the spatial scalability, video at a lower resolution forms part of the base layer. In temporal scalability, the base layer consists of the video sequence coded at a lower frame rate, while in SNR scalability the base layer consists of the video sequence coded at a high quantization step size. In this section we consider two of these scalability modes, temporal and SNR that are relevant to the skip or code mode decision mentioned earlier. We focus our discussion to the use of one enhancement layer. The work described here can be extended to using multiple enhancement layers.

Temporal scalability is achieved by skipping frames while coding the base layer, to obtain a lower frame rate video. Frames that are skipped are predicted (may be forward, backward or bi-directionally predicted) from the current and previous coded frames and the residue and motion vectors are included in the enhancement layer. SNR scalability is achieved by coding frames at a higher quantization step size at the base layer and then coding the residue between these frames and the actual video at a lower quantization step size to form the enhancement layer.

Using the mode decisions described in Sections 3.3 and 3.4, we code a video sequence by sometimes using a high quantization step size and sometimes skipping frames. Hence, this coded video may be viewed as a base layer generated using a encoder that switches between SNR and temporal scalabilities, as detailed later. We can therefore extend our work from the previous section to investigate the error resilience and performance of our strategy when implemented over lossy networks. To do this, we first generate an enhancement layer corresponding to our base layer. This process is highlighted in Figure 3.17 and Figure 3.18.

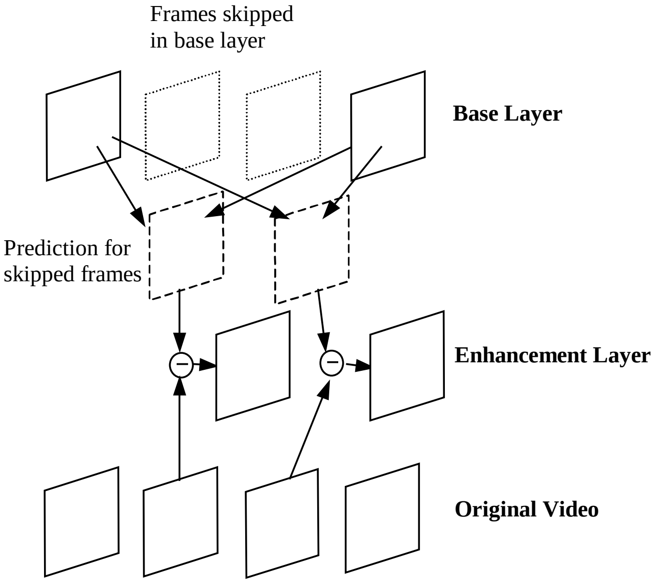

Figure 3.17. Enhancement layer generation for skipped frames

From the figure we can see that when we skip a frame in the base layer, we build a prediction for the frame, that may be forward, backward or bi-directionally predicted from the preceding and following coded frames and the residue between this prediction and the original video is included as part of the enhancement layer. This is equivalent to temporal scalability.

The process of generating the enhancement layer when we code a frame in the base layer is highlighted in Figure 3.18.

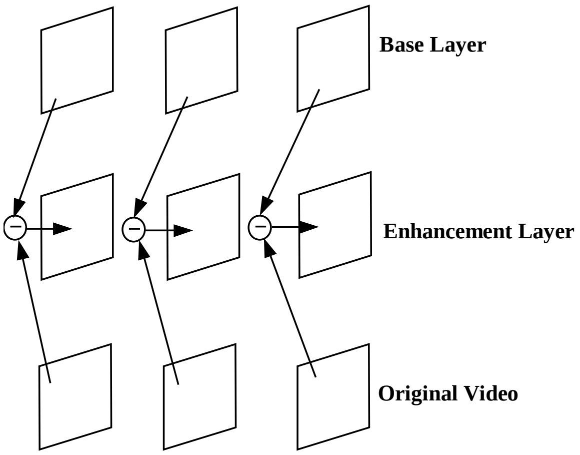

Figure 3.18. Enhancement layer generation for coded frames

Coded frames are subtracted from the original video and the residue for each of these frames is included as part of the enhancement layer, which is coded at a lower quantization step size. This is equivalent to SNR scalability.

Our coding scheme switches between these two modes and we call it adaptive SNR/temporal (AST) scalable coding. Once we have built the enhancement layer corresponding to our base layer, we code it at the same target rate as the base layer. In order to achieve the target rate, we change only the quantization step size and do not allow for skipping of frames. We then simulate lossy network conditions and then reconstruct the video sequence by combining the two layers. The lossy conditions are simulated by throwing away some of the base layer macroblocks and some of the enhancement layer macroblocks and then combining the layers. We examine different error rates and their impact on the performance. Some error concealment is used at the decoder side to improve the quality of the decoded video. When a base layer macroblock is corrupted it is replaced by the corresponding macroblock from the previous frame, while when an enhancement layer macroblock is corrupted, it is thrown away. We also generate an enhancement layer for the No Skip rate control scheme (identical to SNR scalability) and code and combine the layers as before. The resulting rate distortion curves are plotted in Figure 3.19.

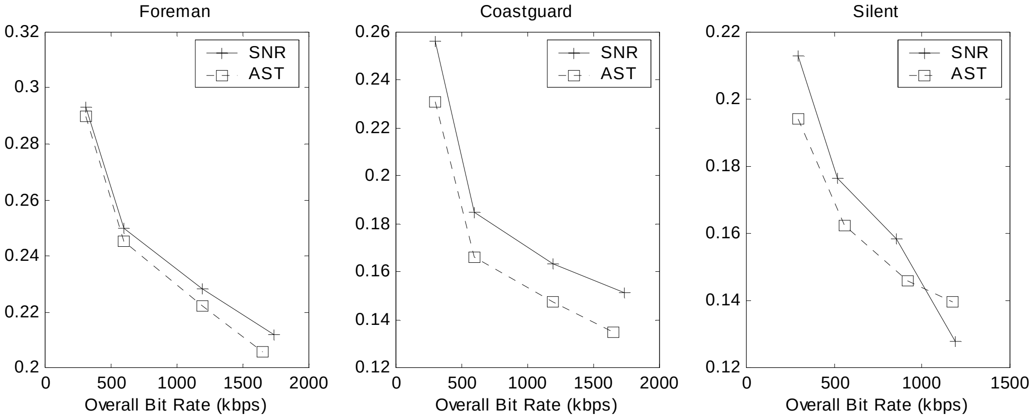

Figure 3.19. Performance under lossy network conditions

The curves plotted in the figure are with 5% loss in the base layer and 10% loss in the enhancement layer. We simulate this lossy environment by randomly discarding 5% of the blocks from the base layer and 10% of the blocks from the enhancement layer. As can be seen from the plots, the performance of the AST is better than using just SNR scalability across different target rates for all the three sequences, with only one exception (Silent sequence at 1200 kbps, 600 kbps for base layer and 600 kbps for enhancement layer). Sample frames from the Foreman sequence to highlight the improvement in distortion are shown in Figure 3.20.

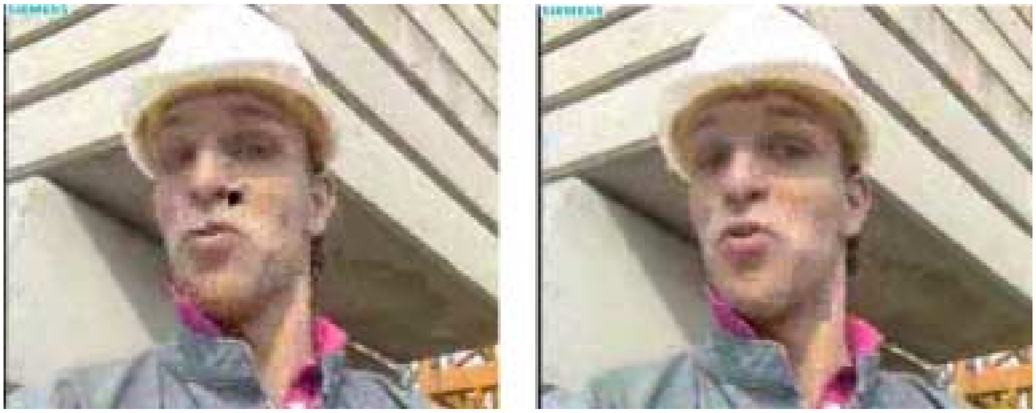

Figure 3.20. Sample frames from Foreman sequence with 5% base loss and 10% enhancement loss SNR Scalability (left) and AST Scalability (right)

The SNR scalability frame has a PSNR of 27.7 dB while the AST scalability frame has a PSNR of 29.09 dB as compared to the original frame.

We also investigate the effect of varying the enhancement layer and base layer losses at a fixed target rate of 300 kbps for base layer and 300 kbps for enhancement layer. We compare the performance with SNR scalability and the resulting improvements are shown in Figure 3.21.

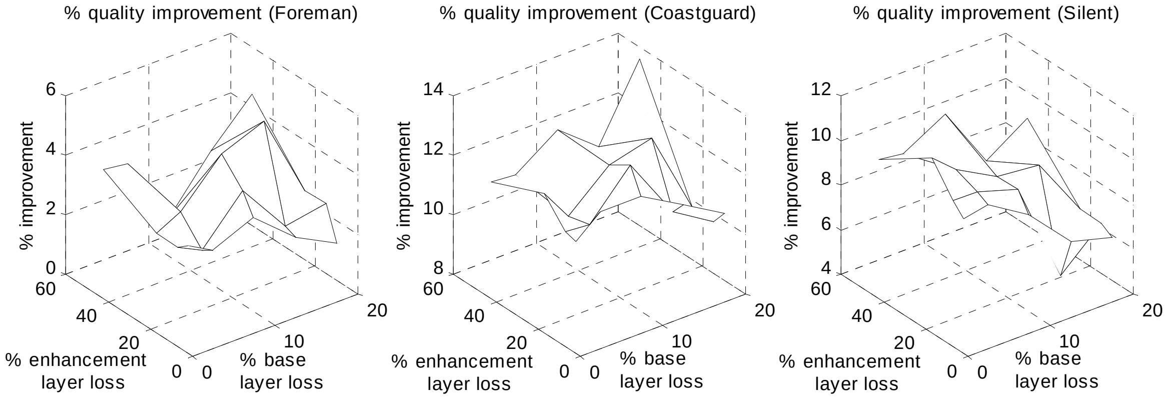

Figure 3.21. Improvement in quality of AST over SNR scalability for different error rates

All of the above plots are generated for a target rate of 300 kbps for the base layer and the same for the enhancement layer. We can see that for all the different error rates the performance in terms of quality is better when we generate the base layer using the classification based mode decision as opposed to just changing the quantization step size. We can also see that the percentage improvement in quality is higher for Silent and Coastguard sequences as opposed to the Foreman sequence. This may be explained by a combination of facts. When we skip a frame, the enhancement layer carries greater amount of information for high motion sequences than it would for a low motion sequence. Also we have a larger amount of losses in the enhancement layer as compared to the base layer. So we lose more information for higher motion sequences when we skip frames in the base layer. Hence we have a smaller improvement for the Foreman sequence. The case with 0% loss in the base layer and 100% loss in the enhancement layer degenerates to the rate control problem that we focused on in Sections 3.3 and 3.4.

3.6. Conclusion

The main contribution of this chapter is the classification based approach to mode decisions in the video encoding process. We successfully convert the problem of minimization of a certain cost function into a standard minimization of classification error problem and use traditional pattern classification techniques to solve it. We use this approach to improve the performance of the Intra-Inter mode decision and reduce the bitstream size by 4.5~4.8% over the mode decision as provided in TMN 10. We then use this approach to the rate control problem and show an improvement in performance in the rate-distortion sense over using the no skip approach both for the instantaneous as well as the look-ahead mode decision. The improvement in quality for the instantaneous decision is 4~12% while for the look-ahead decision it is 7~18% over the no skip rate control. We also extend this work to the scalable video coding and show that with the adaptive SNR/temporal (AST) scalability we improve the performance in terms of quality under error prone conditions by 5~15% over using SNR scalability only. More details on this chapter and the classification based framework for mode decisions may be obtained from our paper in [47].

3.7. Appendices

Appendix A: Correlation coefficient between decision sequence and feature sequence

In order to compute the suitability of features to use in our classifier, we compute the correlation coefficient between the features and the optimal decision sequence. The optimal decision sequence is found using the exhaustive schemes and we view it as a binary sequence of +1s and –1s, corresponding to the two mode decisions. Before we correlate the feature sequence with the decision sequence, we threshold the feature sequence to convert it also to a binary sequence of +1s and –1s. This is done so that we get a better estimate of correlation between the feature sequence and the decision sequence. If we do not convert the feature sequence to a binary sequence, even if the feature is perfectly representative of the decision sequence, i.e. it is high when we have decide Intra (+1) and low when we decide Inter (–1) we do not get a correlation coefficient 1. We highlight this with an example in Figure 3.22.

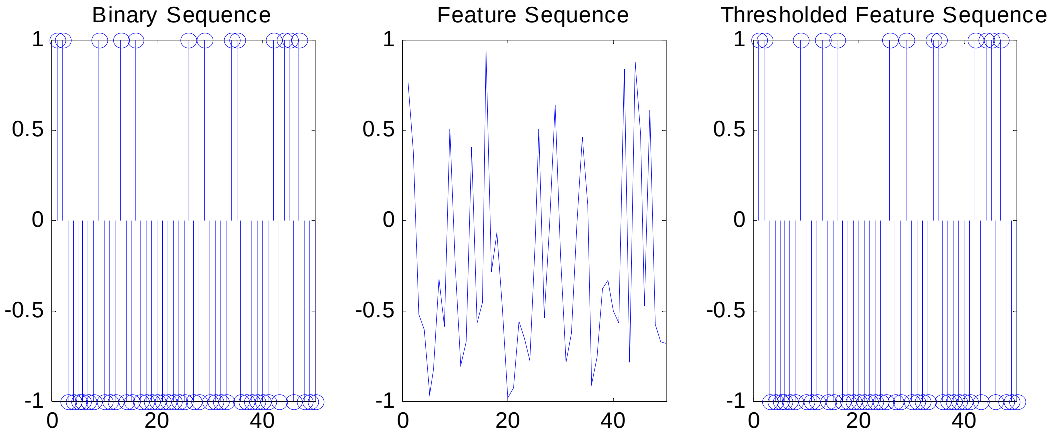

Figure 3.22. Correlation between decision sequence and feature sequence

In this figure each of the three sequences has 50 samples. The feature sequence may be used to predict the decision sequence very precisely, because we can see that when the feature has a high value, the decision sequence has value +1 and when the feature value is low, the decision sequence has value –1. However, when we compute the correlation coefficient between these two sequences, the resulting value is only 0.44. As against this, if we convert the feature sequence to a binary sequence, using a threshold of 0, before computing the correlation coefficient, the resulting value is 1 as desired. We try multiple thresholds to convert every feature sequence into a binary sequence before correlating with the decision sequence and report the best correlation coefficient obtained.

Appendix B: Spatio-Temporal Quality Metric

A quality metric should measure spatial as well as temporal quality. The PSNR provides a poor measure of temporal quality, hence we use the metric proposed by Wolf and Pinson [45]. To evaluate this metric, first the luminance components of the input and output video streams are processed using horizontal and vertical edge enhancement filters. These processed streams are partitioned into spatio-temporal (S-T) regions in which features that quantify spatial activity as a function of angular orientation are extracted. These are then clipped to emulate perceptibility thresholds. Distortions due to gains and losses in feature values are calculated using functional relationships between the input and output feature values that emulate visual masking. These distortions are then collapsed over space and time. The choice of edge enhancement filters and the perceptibility thresholds are optimized based on their correlation with perceptual distortions. The block diagram of this process is shown in Figure 3.23.

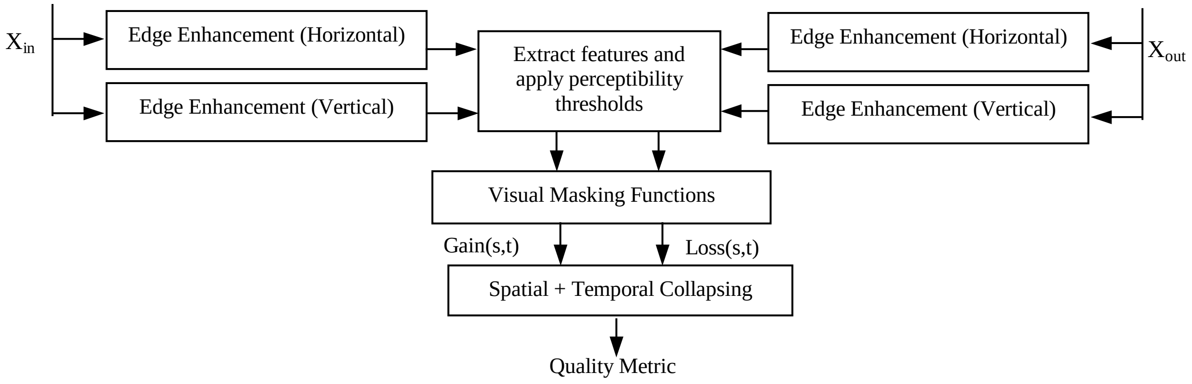
X_{in}
Edge Enhancement (Horizontal)
Edge Enhancement (Vertical)
Extract features and apply perceptibility thresholds
Edge Enhancement (Horizontal)
Edge Enhancement (Vertical)
X_{out}
Visual Masking Functions
Gain(s,t)
Loss(s,t)
Spatial + Temporal Collapsing
Quality Metric

Figure 3.23. Computation of spatio-temporal metric

The quality metric computed as in Figure 3.23 lies in the range [-1,0] with 0 corresponding to perceptually lossless video. We use the negative of this feature as a measure of distortion, with 0 corresponding to no visible distortion and 1 corresponding to a large distortion.

4. Mixtures of Principal Components for Error Concealment

This chapter introduces a stochastic modeling technique called Mixtures of Principal Components to capture data variations efficiently and accurately. This new model for data is then used for error concealment, a decoder post-processing technique to reduce the distortions due to lossy network transmission of video.

Many branches of science are faced with the problem of analyzing high dimensional multi-variate data. Due to lack of visualization tools and also in order to have efficient data processing dimensionality reduction for this multi-variate data becomes critical. Most dimensionality reduction techniques try to reduce or eliminate statistical redundancy between the components of the high-dimensional data to obtain a lower dimensional representation without significant loss of information. There are many linear and non-linear techniques that have been proposed in literature to solve this of dimensionality reduction problem.

Among the linear dimensionality reduction techniques is the Principal Components Analysis (PCA). Given a set of data vectors, PCA identifies the principal directions of variation in the data space. These principal directions of variation correspond to the eigenvectors of the covariance matrix of the data and may be used to represent the data. These eigenvectors may be ranked in the order of importance based on the magnitude of their corresponding eigenvalues, with the eigenvector corresponding to the largest eigenvalue being the most significant one and so on. These eigenvectors may be used as basis vectors to reconstruct the original data vectors. Since we know the principal directions, we may discard some eigenvectors, corresponding to small eigenvalues without incurring a great increase in reconstruction error. This allows us to model the data using a small set of eigenvectors. As mentioned before, for the PCA to represent large variations in the data set, PCA will require larger dimensional representation than would be required by a non-linear modeling technique. However non-linear modeling techniques have problems such as in the lack of ease of computation and the absence of a simple forward backward transformation between the low-dimensional representation and the actual data set.

We would like to use the advantages of using a linear modeling technique, however we would like to extend the PCA to improve the modeling performance for data with large variations. Hence we propose a ***mixture of principal components (MPC)*** to represent the data while optimizing the reconstruction error, however we would like to avoid hard partitioning of the data in order to exploit both the local as well as the global information in the data. Our approach uses a linear combination of reconstructions from component eigenspaces to represent the data. We show an illustration of our approach in Figure 4.1.

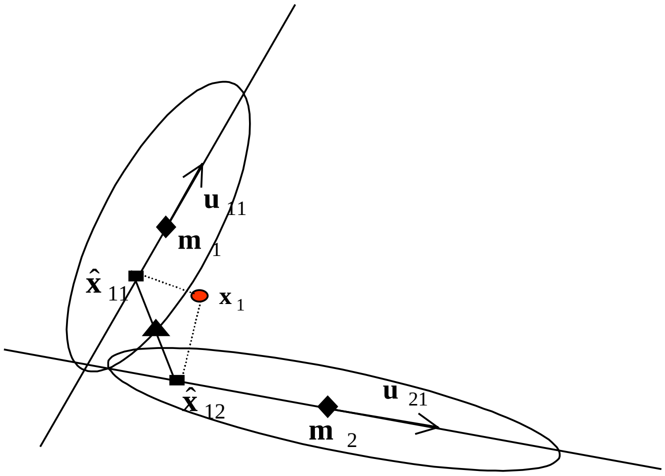

Figure 4.1. Illustration of mixture of eigenspaces

In Figure 4.1 we show data representation and reconstruction using a mixture of two component eigenspaces with one eigenvector each. The original data is distributed as shown by the two ellipses while the eigenspaces have means $\mathbf{m}_1$ and $\mathbf{m}_2$, and eigenvectors $\mathbf{u}_{11}$ and $\mathbf{u}_{21}$ respectively. In the figure, the means are shown as black diamonds and the direction of the eigenvectors is shown as a line passing through the corresponding means. Given a data sample $\mathbf{x}_1$, shown as a dark circle, we first project it onto each of the component eigenspaces to obtain $\hat{\mathbf{x}}_{11}$ and $\hat{\mathbf{x}}_{12}$. We then linearly combine these two projections to obtain the best reconstruction for the data, shown as the dark triangle in the figure.

The problem we are trying to solve is the following. Given a set of data vectors with variations, we would like to automatically train a set (mixture) of eigenspaces so that the reconstruction error is as small is possible. We would like to train all the parameters of these component eigenspaces (the means and the eigenvectors) automatically, as well as weights to linearly combine the individual reconstructions. We formulate this problem of training as a minimum error optimization problem and provide a solution using an iterative Expectation Maximization (EM) kind of algorithm. We iteratively update the weights, the means and the eigenvectors, one by one, ensuring that when we update one of the three, the other two are kept fixed.

Once we provide the solution to the problem, we then highlight the use of these eigenspaces using simulated data and real applications. We first create some random data clusters and try to model them using MPC. We show the improvements over using the PCA and also illustrate some trivial cases for the performance of the MPC. We also include a brief comparison with previously proposed linear extensions to the PCA. Among these extensions are the PPCA and the VQPCA as described in Section 1.2. These extensions to the PCA, including the MPC, can be classified into those that use hard or soft clustering techniques and those that are optimized in terms of the squared error or the likelihood of observing the data. We group these appropriately in Table 4-1.

Table 4-1. Linear Extensions to the PCA

	Optimize Likelihood of Observing Data	**Optimize Reconstruction Error**
Hard Clustering		VQPCA
Soft Clustering	PPCA	MPC

As may be seen from Table 4-1, PPCA uses soft clustering of data while optimizing the likelihood of observing data, while VQPCA uses hard clustering and optimizes the reconstruction error. The MPC uses soft clustering of the data while trying to optimize the reconstruction error. It is known from literature that the reconstruction error performance of the VQPCA is better than the reconstruction error performance of the PPCA. This is to be expected, as the PPCA is not designed to optimize the reconstruction error. In our work we are also attempting to optimize the reconstruction error, so we also include a brief comparison with the VQPCA and show that for sample random data the MPC has better error performance.

We then focus on using this model for error concealment of video sequences with losses due to transmission over networks. Loss of compressed video information during transmission leads to objectionable visual distortion in the reconstructed video. Hence it is necessary for the decoder to perform error concealment to minimize this distortion. Some previously described error concealment schemes use spatial domain interpolation, projection onto convex sets (POCS) and temporal domain interpolation, such as interpolating the motion field. We propose to use model based error concealment. Given a region of interest in the video, we build a model for the data using the MPC and then use this model to replenish any missing data. We evaluate the performance of the scheme in terms of the PSNR as compared to the error free video sequence and highlight improvements over the currently used error concealment schemes.

In order to illustrate the improvements to be gained by capturing data variations efficiently we use the MPC for the task of face recognition. We examine face data with pose and lighting variations and use both the PCA and the MPC to capture the data variations to model each of the subjects in the database. We then use these models for recognition and show that using the MPC leads to a significant improvement in the recognition performance. We also briefly examine a face tracking problem that is difficult for conventional trackers due to extreme lighting variations. We propose a scheme to model the background using the MPC to capture the large lighting variations. The model may be used to reconstruct the background given a test image and we can obtain a foreground map by subtracting the background from the test image. This foreground map may then be used by the tracker to track the face. We illustrate this approach using data collected in a car environment.

This chapter is organized as follows. Section 4.1 describes the notation used in this report. We provide the formulation of the problem in terms of an optimization criterion in Section 4.2, where we also describe the iterative approach to obtaining the solution. Sections 4.3, 4.4 and 4.5 include the derivations for the solutions for the weights, the means and the eigenvectors. Section 4.6 relates this derivation to the PCA, considering the case when we have only one mixture component. We then include some simulation results in Section 4.7. We use the MPC for error concealment in Section 4.8. We use the MPC for the task of face recognition in Section 4.9. We also describe our face tracking work in Section 4.10. We conclude in Section 4.11.

4.1. Notation

N : Number of data points
D : Dimension of data vectors
M : Number of mixture components
P : Number of eigenvectors in each mixture component
$\mathbf{x}_i$: Data vectors $(i = 1 \ldots N)$
$\hat{\mathbf{x}}_{ij}$: Reconstruction for vector i from mixture j $(i = 1 \ldots N, j = 1 \ldots M)$
$\hat{\mathbf{X}}_i$: Matrix with M reconstructions ($\hat{\mathbf{x}}_{ij}$) as columns $(D \times M)$
$\mathbf{m}_j$: Mean of the j^{th} mixture component
$\mathbf{u}_{jk}$: k^{th} eigenvector of j^{th} mixture component $(j = 1 \ldots M, k = 1 \ldots P)$
$\mathbf{U}_j$: Eigenvector matrix for j^{th} mixture component $(D \times P)$
$\mathbf{w}_i$: Weight vector to combine M reconstructions for vector i. It has elements w_{ij} $(j = 1 \ldots M)$

4.2. Optimization Criterion

Given a set of data vectors, we are attempting to represent them using a mixture containing M eigenspaces, each of which has P eigenvectors. We need to find the means and the eigenvectors for each of these mixture components and also we need to find the set of weights to combine the reconstructions from each of these components. We attempt to find these by minimizing the mean squared error between the data vectors and the final reconstruction. This problem of minimizing the squared error through the choice of the means, the sets of eigenvectors and the set of weights, may be mathematically written as in equation (4.1).

$$\min_{\mathbf{w}_i, \mathbf{m}_j, \mathbf{u}_{jk}} \frac{1}{N} \sum_{i=1}^{N} \left\| \mathbf{x}_i - \sum_{j=1}^{M} w_{ij} \left[\underbrace{\mathbf{m}_j + \sum_{k=1}^{P} \left[(\mathbf{x}_i - \mathbf{m}_j)^T \mathbf{u}_{jk} \right] \mathbf{u}_{jk}}_{\hat{\mathbf{x}}_{ij}} \right] \right\|^2 \tag{4.1}$$

The approach we adopt is an iterative one, similar to the EM algorithm for Gaussian mixture training. The solution is iterative due to the fact that we cannot find an analytical solution for the means, the weights and the eigenvectors. We solve for the weights and the means in terms of the eigenvectors and substitute these values into the optimization criterion, to obtain an analytical solution for the eigenvectors. However, we then need to solve an equation with order greater than four to obtain the solution for the eigenvectors. It is known from literature that a closed form solution for such equations cannot be obtained. Hence we use an iterative approach to obtain the solution. We first initialize the weights, the means and the eigenvectors for the different components randomly. We first optimize the weights for each data vector individually, following which we optimize the means of the component eigenspaces and finally optimize the eigenvectors. During the process of optimizing one set of parameters, (the weights, the means and the eigenvectors), all the other parameters are fixed, similar to the EM training. This process is repeated till convergence (parameters do not change by more than a threshold).

4.3. Solution for the weights

The weights are solved for individually for each of the vectors. So from our optimization criterion, we may drop the summation over all vectors. The only constraint on the weights is that we impose on the weights is that they are required to sum to one. We may rewrite the optimization criterion in (4.1) as follows.

$$\min_{\mathbf{w}_i} \left\| \mathbf{x}_i - \sum_{j=1}^{M} w_{ij} \hat{\mathbf{x}}_{ij} \right\|^2 = \min_{\mathbf{w}_i} \left\| \mathbf{x}_i - \hat{\mathbf{X}}_i \mathbf{w}_i \right\|^2 \tag{4.2}$$

since we are weighting and summing each of the mixture component reconstructions (columns of the matrix $\hat{\mathbf{X}}_i$) with weights w_{ij} (elements of weight vector $\mathbf{w}_i$), to get as close to the original vector as possible.

Adding in the constraint using the Lagrange multiplier, equation (4.2) may be rewritten as below.

$$\min_{\mathbf{w}_i,\lambda} \left[\left(\mathbf{x}_i - \hat{\mathbf{X}}_i \mathbf{w}_i \right)^T \left(\mathbf{x}_i - \hat{\mathbf{X}}_i \mathbf{w}_i \right) + \lambda \left(\mathbf{w}_i^T \mathbf{one} - 1 \right) \right] \text{where } \mathbf{one} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} (M \times 1) \tag{4.3}$$

Taking derivatives with respect to $\mathbf{w}_i$ and λ and setting the result to zero we get the following equations

$$\begin{aligned} 2\hat{\mathbf{X}}_i^T \hat{\mathbf{X}}_i \mathbf{w}_i - 2\hat{\mathbf{X}}_i^T \mathbf{x}_i + \lambda \mathbf{one} = 0 \\ \mathbf{w}_i^T \mathbf{one} = \mathbf{one}^T \mathbf{w}_i = 1 \end{aligned} \tag{4.4}$$

The two equations in (4.4) may be grouped together as follows

$$\begin{bmatrix} 2\hat{\mathbf{X}}_i^T \hat{\mathbf{X}}_i & \mathbf{one} \\ \mathbf{one}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{w}_i \\ \lambda \end{bmatrix} = \begin{bmatrix} 2\hat{\mathbf{X}}_i^T \mathbf{x}_i \\ 1 \end{bmatrix} \tag{4.5}$$

We can thus solve for the weights using equation (4.5).

4.4. Solutions for the means

While solving for the means we now fix the weights and the eigenvectors. We may rewrite equation (4.1) as follows

$$\min_{\mathbf{m}_q} \frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{x}_i - \hat{\mathbf{X}}_i\mathbf{w}_i\right\|^2 \qquad (4.6)$$

We need to now expand $\hat{\mathbf{X}}_i$ in terms of the means before we can take derivatives and set to zero.

$$\hat{\mathbf{x}}_{ij} = \mathbf{U}_j\mathbf{U}_j^T\left(\mathbf{x}_i - \mathbf{m}_j\right) + \mathbf{m}_j = \left(\underbrace{\mathbf{I}_D - \mathbf{U}_j\mathbf{U}_j^T}_{\mathbf{A}_j}\right)\mathbf{m}_j + \underbrace{\mathbf{U}_j\mathbf{U}_j^T}_{\mathbf{B}_j}\mathbf{x}_i$$

$$\hat{\mathbf{X}}_i = \sum_{j=1}^{M}\hat{\mathbf{x}}_{ij}\mathbf{e}_j^T \text{ where } \mathbf{e}_j = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} j^{th}\text{position} \qquad (4.7)$$

$$\text{Note : Since } \mathbf{U}_j^T\mathbf{U}_j = \mathbf{I}_M \Rightarrow \mathbf{A}_j^T\mathbf{A}_j = \mathbf{A}_j \ \& \ \mathbf{A}_j^T\mathbf{B}_j = \mathbf{0}$$

In the above equation $\mathbf{B}_j$ is the matrix that projects data onto the sub-space spanned by the eigenvectors in $\mathbf{U}_j$, while $\mathbf{A}_j$ is the matrix that projects data onto the space orthogonal to this sub-space. We may use equation (4.7) to replace $\hat{\mathbf{x}}_i$ in equation (4.6) and hence, the optimization criterion may be rewritten as follows.

$$\min_{\mathbf{m}_q} \frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{x}_i - \left[\sum_{j=1}^{M}\left(\mathbf{A}_j\mathbf{m}_j + \mathbf{B}_j\mathbf{x}_i\right)\mathbf{e}_j^T\right]\mathbf{w}_i\right\|^2 \qquad (4.8)$$

$$\text{where } \left\|\bullet\right\|^2 = \left(\bullet\right)^T\left(\bullet\right)$$

We may now expand equation (4.8) into individual terms and rewrite it as follows.

$$\min_{\mathbf{m}_q} \frac{1}{N}\sum_{i=1}^{N}\left[\begin{array}{l} \mathbf{x}_i^T\mathbf{x}_i - 2\mathbf{w}_i^T\left[\sum_{j=1}^{M}\left(\mathbf{A}_j\mathbf{m}_j + \mathbf{B}_j\mathbf{x}_i\right)\mathbf{e}_j^T\right]^T\mathbf{x}_i + \\ \mathbf{w}_i^T\left[\sum_{j=1}^{M}\left(\mathbf{A}_j\mathbf{m}_j + \mathbf{B}_j\mathbf{x}_i\right)\mathbf{e}_j^T\right]^T\left[\sum_{k=1}^{M}\left(\mathbf{A}_k\mathbf{m}_k + \mathbf{B}_k\mathbf{x}_i\right)\mathbf{e}_k^T\right]\mathbf{w}_i \end{array}\right] \qquad (4.9)$$

We now drop the terms independent of $\mathbf{m}_j$ and expand the inner product terms.

$$\min_{\mathbf{m}_q} \frac{1}{N} \sum_{i=1}^{N} \begin{bmatrix} -2\sum_{j=1}^{M} \mathbf{w}_i^T \mathbf{e}_j \mathbf{m}_j^T \mathbf{A}_j^T \mathbf{x}_i + \sum_{j=1}^{M}\sum_{k=1}^{M} \mathbf{w}_i^T \mathbf{e}_j \mathbf{m}_j^T \mathbf{A}_j^T \mathbf{A}_k \mathbf{m}_k \mathbf{e}_k^T \mathbf{w}_i \\ \sum_{j=1}^{M}\sum_{k=1}^{M} \mathbf{w}_i^T \mathbf{e}_j \mathbf{m}_j^T \mathbf{A}_j^T \mathbf{B}_k \mathbf{x}_i \mathbf{e}_k^T \mathbf{w}_i + \sum_{j=1}^{M}\sum_{k=1}^{M} \mathbf{w}_i^T \mathbf{e}_j \mathbf{x}_i^T \mathbf{B}_j^T \mathbf{A}_k \mathbf{m}_k \mathbf{e}_k^T \mathbf{w}_i \end{bmatrix} \tag{4.10}$$

Notice that $\mathbf{w}_i^T \mathbf{e}_j = \mathbf{e}_j^T \mathbf{w}_i = \mathbf{w}_{ij}$, which are scalars and so may be moved to the front of each term. We now take derivatives with respect to $\mathbf{m}_q$ and set the result to zero.

$$\sum_{i=1}^{N} \begin{bmatrix} -2w_{iq}\mathbf{A}_q^T \mathbf{x}_i + 2\sum_{j=1,\, j\neq q}^{M} w_{iq} w_{ij} \mathbf{A}_q^T \mathbf{A}_j \mathbf{m}_j + \\ 2w_{iq}^2 \mathbf{A}_q^T \mathbf{A}_q \mathbf{m}_q + 2\sum_{k=1}^{M} w_{iq} w_{ik} \mathbf{A}_q^T \mathbf{B}_k \mathbf{x}_i \end{bmatrix} = \mathbf{0} \tag{4.11}$$

Now we may use the properties that $\mathbf{U}_j^T \mathbf{U}_j = \mathbf{I}_M \Rightarrow \mathbf{A}_j^T \mathbf{A}_j = \mathbf{A}_j \;\&\; \mathbf{A}_j^T \mathbf{B}_j = \mathbf{0}$ to simplify equation (4.11) as follows.

$$\mathbf{A}_q^T \mathbf{m}_q = \frac{1}{\sum_{i=1}^{N} w_{iq}^2} \mathbf{A}_q^T \left[\sum_{i=1}^{N} w_{iq} \left(\mathbf{x}_i - \sum_{j=1,\, j\neq q}^{M} w_{ij} \hat{\mathbf{x}}_{ij} \right) \right] \tag{4.12}$$

Equation (4.12) has multiple solutions due to the fact that $\mathbf{A}_j$ is singular. One solution for the means is as follows.

$$\mathbf{m}_q = \frac{1}{\sum_{i=1}^{N} w_{iq}^2} \left[\sum_{i=1}^{N} w_{iq} \left(\mathbf{x}_i - \sum_{j=1,\, j\neq q}^{M} w_{ij} \hat{\mathbf{x}}_{ij} \right) \right] \tag{4.13}$$

Clearly, any vector orthogonal to $\mathbf{A}_q$ may be added to $\mathbf{m}_q$ as defined in equation (4.13) to still remain a valid solution to equation (4.12). Thus, the means are allowed to move within the sub-space (hyper-plane) spanned by the eigenvectors in $\mathbf{U}_q$. This non-uniqueness does not affect the reconstruction error, as it is measured as a distance from the projection onto the hyper-plane, which is independent of the location of the mean within this hyper-plane. As an illustration we show a simple example scenario in Figure 4.2.

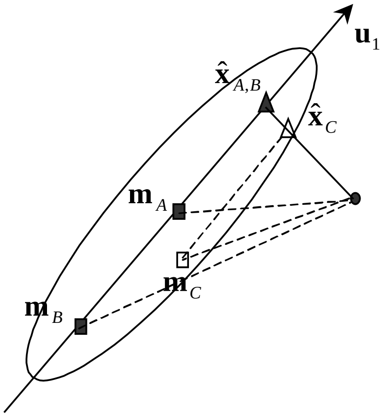

Figure 4.2. Projection onto eigenspace as function of mean location

In the example scenario we consider a single eigenspace with one principal direction $\mathbf{u}_1$, shown in the figure as the principal axis of the ellipse. When we project data point $\mathbf{x}$ onto this eigenspace, whether the mean is located at position $\mathbf{m}_A$ or at position $\mathbf{m}_B$, the resulting projection is $\hat{\mathbf{x}}_{A,B}$. In fact this is always the resulting projection as long as the location of the mean changes along the line specified by $\mathbf{u}_1$ passing through $\mathbf{m}_A$. As an illustration when the mean moves to location $\mathbf{m}_C$, the resulting projection also changes to $\hat{\mathbf{x}}_C$, which is different from $\hat{\mathbf{x}}_{A,B}$, thereby leading to a different reconstruction error. Hence we can tolerate this non-uniqueness of the solution for the mean, as long as it is limited to within the hyper-plane spanned by the eigenvectors. For our training we use the solution for the means from equation (4.13) as it is the minimum norm solution for the mean.

4.5. Solution for Eigenvectors

We may now fix the weights and the means. The optimization criterion from equation (4.1) may be rewritten below in equation (4.14).

$$\min_{\mathbf{u}_{jk}} \frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{x}_i - \sum_{j=1}^{M} w_{ij}\left[\mathbf{m}_j + \sum_{k=1}^{P}\left[\left(\mathbf{x}_i - \mathbf{m}_j\right)^T \mathbf{u}_{jk}\right]\mathbf{u}_{jk}\right]\right\|^2 \tag{4.14}$$

Let us define $y_{ijk} = \left(\mathbf{x}_i - \mathbf{m}_j\right)^T \mathbf{u}_{jk}$. Using this, the minimization criterion of (4.14) may be written as follows.

$$\min_{\mathbf{u}_{rs}} \frac{1}{N}\sum_{i=1}^{N}\left[\begin{array}{l} \mathbf{x}_i^T\mathbf{x}_i - 2\mathbf{x}_i^T\sum_{j=1}^{M} w_{ij}\left(\mathbf{m}_j + \sum_{k=1}^{P} y_{ijk}\mathbf{u}_{jk}\right) + \\ \sum_{j=1}^{M}\sum_{a=1}^{M} w_{ij}w_{ia}\left(\mathbf{m}_j + \sum_{k=1}^{P} y_{ijk}\mathbf{u}_{jk}\right)^T\left(\mathbf{m}_a + \sum_{b=1}^{P} y_{iab}\mathbf{u}_{ab}\right) \end{array}\right] \tag{4.15}$$

We now expand the terms in equation (4.15) and get the following.

$$\min_{\mathbf{u}_{rs}} \frac{1}{N}\sum_{i=1}^{N}\left[\begin{array}{l} \mathbf{x}_i^T\mathbf{x}_i - 2\mathbf{x}_i^T\sum_{j=1}^{M} w_{ij}\left(\mathbf{m}_j + \sum_{k=1}^{P} y_{ijk}\mathbf{u}_{jk}\right) + \sum_{j=1}^{M}\sum_{a=1}^{M} w_{ij}w_{ia}\mathbf{m}_j^T\mathbf{m}_a + \\ 2\sum_{j=1}^{M}\sum_{a=1}^{M} w_{ij}w_{ia}\mathbf{m}_j^T\sum_{b=1}^{P} y_{iab}\mathbf{u}_{ab} + \sum_{j=1}^{M}\sum_{a=1}^{M} w_{ij}w_{ia}\sum_{k=1}^{P}\sum_{b=1}^{P} y_{ijk}y_{iab}\mathbf{u}_{jk}^T\mathbf{u}_{ab} \end{array}\right] \tag{4.16}$$

Now we may drop all terms independent of $\mathbf{u}_{jk}$ and rewrite equation (4.16) using the knowledge that all eigenvectors of one mixture component are orthogonal to each other. (Note: Eigenvectors from different mixture components are not constrained to be orthogonal).

$$\min_{\mathbf{u}_{rs}} \frac{1}{N}\sum_{i=1}^{N}\left[\begin{array}{l} -2\sum_{j=1}^{M} w_{ij}\sum_{k=1}^{P}\mathbf{u}_{jk}^T\left(\mathbf{x}_i - \mathbf{m}_j\right)\mathbf{x}_i^T\mathbf{u}_{jk} + 2\sum_{j=1}^{M}\sum_{a=1}^{M} w_{ij}w_{ia}\sum_{b=1}^{P}\mathbf{u}_{ab}^T\left(\mathbf{x}_i - \mathbf{m}_a\right)\mathbf{m}_j^T\mathbf{u}_{ab} + \\ \sum_{j=1}^{M}\sum_{a=1,a\neq j}^{M} w_{ij}w_{ia}\sum_{k=1}^{P}\sum_{b=1}^{P} y_{ijk}\mathbf{u}_{ab}^T\left(\mathbf{x}_i - \mathbf{m}_a\right)\mathbf{u}_{jk}^T\mathbf{u}_{ab} + \\ \sum_{j=1}^{M} w_{ij}^2\sum_{k=1}^{P}\mathbf{u}_{jk}^T\left(\mathbf{x}_i - \mathbf{m}_j\right)\left(\mathbf{x}_i - \mathbf{m}_j\right)^T\mathbf{u}_{jk} \end{array}\right] \tag{4.17}$$

We also need to add the constraint that the eigenvectors need to have a norm of 1, and we may use the Lagrange multiplier method to add in the constraint to obtain equation (4.18).

$$\min_{\mathbf{u}_{rs}} \frac{1}{N}\sum_{i=1}^{N}\left[\begin{array}{l} -2\sum_{j=1}^{M} w_{ij}\sum_{k=1}^{P}\mathbf{u}_{jk}^T\left(\mathbf{x}_i - \mathbf{m}_j\right)\mathbf{x}_i^T\mathbf{u}_{jk} + \\ 2\sum_{j=1}^{M}\sum_{a=1}^{M} w_{ij}w_{ia}\sum_{b=1}^{P}\mathbf{u}_{ab}^T\left(\mathbf{x}_i - \mathbf{m}_a\right)\mathbf{m}_j^T\mathbf{u}_{ab} + \\ \sum_{j=1}^{M}\sum_{a=1,a\neq j}^{M} w_{ij}w_{ia}\sum_{k=1}^{P}\sum_{b=1}^{P} y_{ijk}\mathbf{u}_{ab}^T\left(\mathbf{x}_i - \mathbf{m}_a\right)\mathbf{u}_{jk}^T\mathbf{u}_{ab} + \\ \sum_{j=1}^{M} w_{ij}^2\sum_{k=1}^{P}\mathbf{u}_{jk}^T\left(\mathbf{x}_i - \mathbf{m}_j\right)\left(\mathbf{x}_i - \mathbf{m}_j\right)^T\mathbf{u}_{jk} \end{array}\right] + \lambda\left(\mathbf{u}_{rs}^T\mathbf{u}_{rs} - 1\right) \tag{4.18}$$

We may now take derivative with respect to $\mathbf{u}_{rs}$ and set the result to zero to obtain equation (4.19). Note that the term in the third line of equation (4.18) has two terms containing $\mathbf{u}_{rs}$, one when $a = r \,\&\, b = s$ and one when $j = r \,\&\, k = s$, which do not happen together due to the

$a \neq j$ condition. Both these terms are identical and their derivative may be grouped together as shown below in equation (4.19).

$$\frac{1}{N}\sum_{i=1}^{N}\left[\begin{array}{l} -2w_{ir}\left[(\mathbf{x}_i-\mathbf{m}_r)\mathbf{x}_i^T+\mathbf{x}_i(\mathbf{x}_i-\mathbf{m}_r)^T\right]\mathbf{u}_{rs}+ \\ 2\sum_{j=1}^{M}w_{ij}w_{ir}\left[(\mathbf{x}_i-\mathbf{m}_r)\mathbf{m}_j^T+\mathbf{m}_j(\mathbf{x}_i-\mathbf{m}_r)^T\right]\mathbf{u}_{rs}+ \\ 2\sum_{j=1,\,j\neq r}^{M}w_{ij}w_{ir}\sum_{k=1}^{P}\left[\mathbf{u}_{jk}^T(\mathbf{x}_i-\mathbf{m}_j)\right]\left[(\mathbf{x}_i-\mathbf{m}_r)\mathbf{u}_{jk}^T+\mathbf{u}_{jk}(\mathbf{x}_i-\mathbf{m}_r)^T\right]\mathbf{u}_{rs}+ \\ 2w_{ir}^2(\mathbf{x}_i-\mathbf{m}_r)(\mathbf{x}_i-\mathbf{m}_r)^T\mathbf{u}_{rs} \end{array}\right]=-2\lambda\mathbf{u}_{rs} \quad (4.19)$$

This is clearly an eigenvector, eigenvalue problem that may be written as $\mathbf{C}_r\mathbf{u}_{rs}=\lambda\mathbf{u}_{rs}$ where the matrix $\mathbf{C}_r$ is defined as follows.

$$\mathbf{C}_r=\frac{1}{N}\sum_{i=1}^{N}\left[\begin{array}{l} w_{ir}\left[(\mathbf{x}_i-\mathbf{m}_r)\mathbf{x}_i^T+\mathbf{x}_i(\mathbf{x}_i-\mathbf{m}_r)^T\right]- \\ \sum_{j=1}^{M}w_{ij}w_{ir}\left[(\mathbf{x}_i-\mathbf{m}_r)\mathbf{m}_j^T+\mathbf{m}_j(\mathbf{x}_i-\mathbf{m}_r)^T\right]- \\ \sum_{j=1,\,j\neq r}^{M}w_{ij}w_{ir}\sum_{k=1}^{P}\left[\mathbf{u}_{jk}^T(\mathbf{x}_i-\mathbf{m}_j)\right]\left[(\mathbf{x}_i-\mathbf{m}_r)\mathbf{u}_{jk}^T+\mathbf{u}_{jk}(\mathbf{x}_i-\mathbf{m}_r)^T\right]- \\ w_{ir}^2(\mathbf{x}_i-\mathbf{m}_r)(\mathbf{x}_i-\mathbf{m}_r)^T \end{array}\right] \quad (4.20)$$

The first P eigenvectors of this matrix are the desired eigenvectors of the mixture component.

4.6. Simplification for *M*=1

In order to verify the correctness of the derivation we consider the case when the number of mixture components is one, when our derivation should be identical to the PCA. We first examine the mean. The mean may be obtained as in equation (4.13). However, when we have only one component, we can make a lot of simplifications and these are shown below.

$$\mathbf{m}_q=\frac{1}{\underset{N}{\cancel{\sum_{i=1}^{N}w_{iq}^2}}}\left[\sum_{i=1}^{N}\cancel{w}_{iq}\left(\mathbf{x}_i-\cancel{\sum_{j=1,\,j\neq q}^{M}w_{ij}\hat{\mathbf{x}}_{ij}}\right)\right]$$

All the weights are set to one, since there is only one component, and then the term consisting of reconstructions from other mixture components may be removed. This means that the mean may be obtained as shown below.

$$\mathbf{m}_q = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i \tag{4.21}$$

This is clearly the sampled mean, which is identical to the PCA. After examining the mean, we now examine the matrix $\mathbf{C}_r$ defined as in equation (4.20).

$$\mathbf{C}_r = \frac{1}{N}\sum_{i=1}^{N}\begin{bmatrix} \cancel{w_{ir}}\left[(\mathbf{x}_i - \mathbf{m}_r)\mathbf{x}_i^T + \mathbf{x}_i(\mathbf{x}_i - \mathbf{m}_r)^T\right] - \\ \cancel{\sum_{j=1}^{M} w_{ij} w_{ir}}\left[(\mathbf{x}_i - \mathbf{m}_r)\mathbf{m}_j^T + \mathbf{m}_j(\mathbf{x}_i - \mathbf{m}_r)^T\right] - \\ \cancel{\sum_{j=1, j\neq r}^{M} w_{ij} w_{ir}\sum_{k=1}^{P}\left[\mathbf{u}_{jk}^T(\mathbf{x}_i - \mathbf{m}_j)\right]\left[(\mathbf{x}_i - \mathbf{m}_r)\mathbf{u}_{jk}^T + \mathbf{u}_{jk}(\mathbf{x}_i - \mathbf{m}_r)^T\right]} \\ \cancel{w_{ir}^2}(\mathbf{x}_i - \mathbf{m}_r)(\mathbf{x}_i - \mathbf{m}_r)^T \end{bmatrix}$$

Since there is only one mixture component, the third term vanishes, all the weights are set to one and the summation over the number of mixtures may be replaced with the single term with $j = r = 1$. Using this simplification, the matrix $\mathbf{C}_r$ may be rewritten as below.

$$\mathbf{C}_r = \frac{1}{N}\sum_{i=1}^{N}\left[(\mathbf{x}_i - \mathbf{m}_r)(\mathbf{x}_i - \mathbf{m}_r)^T\right] \tag{4.22}$$

Clearly, this is identical to the sample covariance matrix for the PCA.

4.7. Simulation Results

In order to evaluate the performance of the MPC representation, we create some sample test data. One issue that needs to be addressed before modeling the data is the choice of the number of mixture components *M* and the number of eigenvectors *P* per component. Currently we do not have an analytical solution to determine these numbers, and they are determined empirically. The number of mixture components *M* is incremented while measuring the performance of the model in terms of the task at hand, until we realize that the improvement in performance is marginal, defined using a threshold. For any given *M*, the number of eigenvectors *P*, is chosen so that we capture 90% of the energy of the original data.

We first create 2-D data uniformly distributed in a ring and then try to model the data using two mixtures with one eigenvector each. The resulting means and eigenvectors are superimposed on the data and are shown in Figure 4.3.

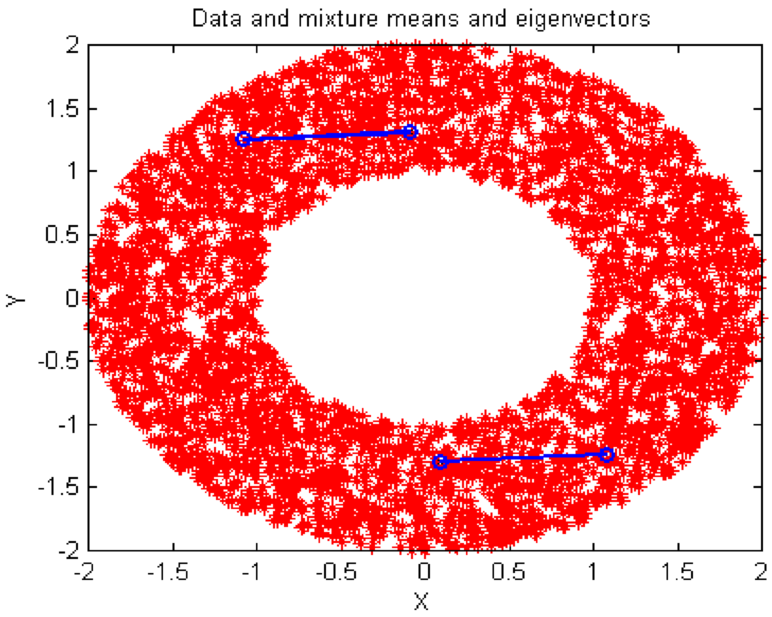

Figure 4.3. Means and eigenvectors for ring data

From Figure 4.3 we see that the two components have eigenvectors that are parallel to each other, with means diagonally across the ring. This is to be expected; due to the way we define the final reconstruction (linear combination of the individual reconstructions). Any point in the 2-D space may be reconstructed without any error using projections onto two parallel lines and interpolating them using weights. This leads to there being no error in the data representation. Such trivial cases occur when the number of mixture components equals or exceeds the dimension of the data to be represented. However such cases are not commonly encountered in practice, as typically the number of mixture components used is much smaller than the data dimension. We show the squared error for this case in Figure 4.4.

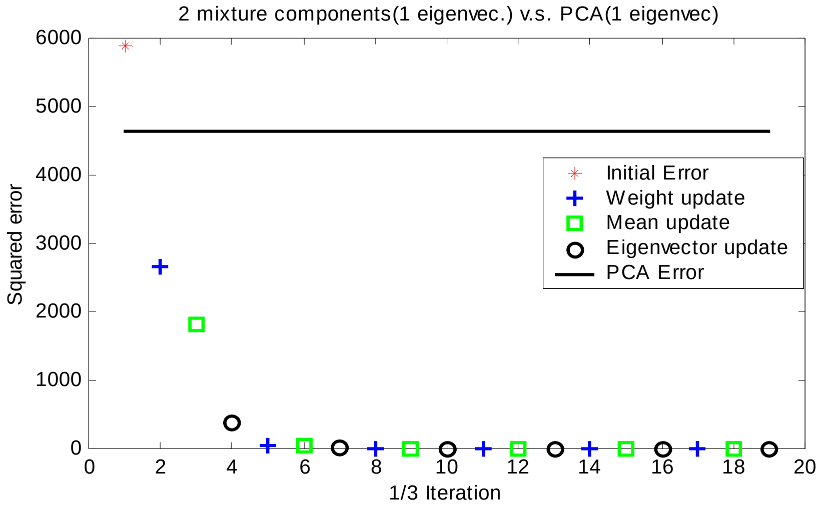

Figure 4.4. Squared error for 2-D ring data

In Figure 4.4, we plot the squared error as a function of the number of iterations. We show the intermediate iteration steps, after updating the weights (plus), the means (square) and the eigenvectors (circle). We start with a random initialization and the corresponding error is plotted using a star. As can be seen the squared error for this MPC goes to zero. We also include the error for PCA with one eigenvector, shown as a straight line on the plot.

We next create a set of 3-D data, distributed in two clusters and try to model the data using two mixture components with one eigenvector each. We create 3-D data clusters as they are easy to visualize and may be shown on a plot. The results of the training are shown in Figure 4.5.

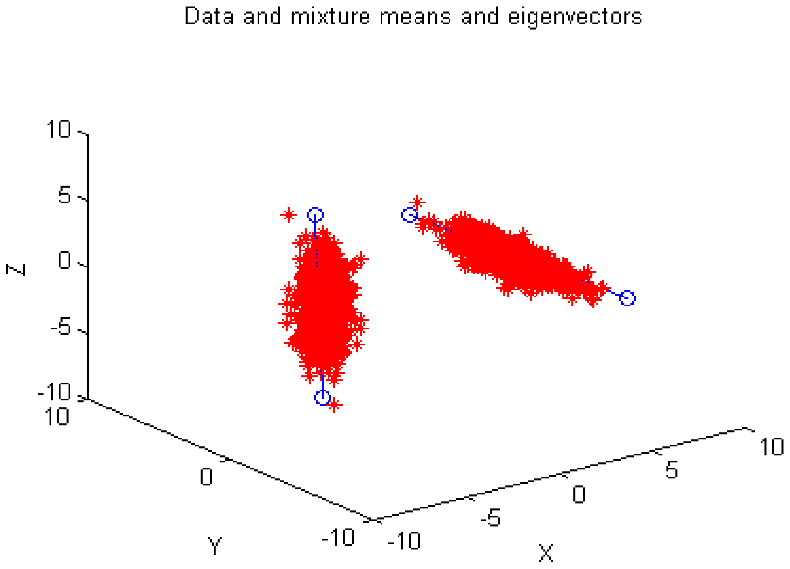

Figure 4.5. 3-D data with mixture means and eigenvectors

In the plot, the data vectors are plotted as stars and on the data we superimpose lines parallel to the two mixture component eigenvectors, and passing through their respective means. As can be seen, the two means of the mixture components converge to the center of the clusters, with the eigenvectors capturing the principal direction of the cluster. We also try to model the data using PCA with two eigenvectors and the resulting squared error is shown in Figure 4.6.

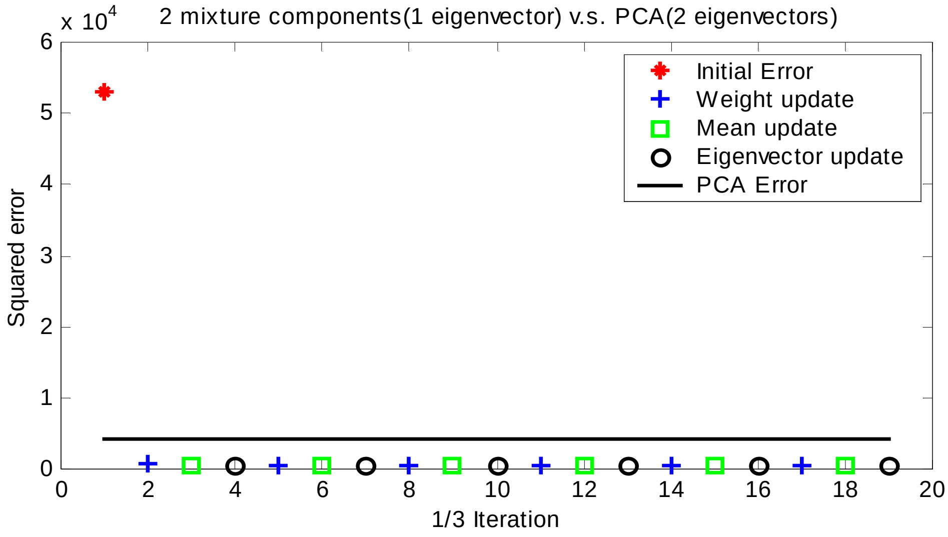

Figure 4.6. Squared error for 3-D data in two clusters

We can see from Figure 4.6 that the error for the MPC with two components with one eigenvector each is much smaller than for PCA with two eigenvectors also. In fact even though the PCA has the same number of total eigenvectors as the MPC, the MPC has around 8 times smaller squared error, leading to a more efficient representation. This shows that the MPC is more efficient than the PCA to represent data that consists of multiple clusters.

For this same random data we also train the multiple clusters using the VQPCA algorithm. In order to make a fair comparison we train two clusters with one eigenvector each, as for the MPC. The resulting means and eigenvectors are shown in Figure 4.7.

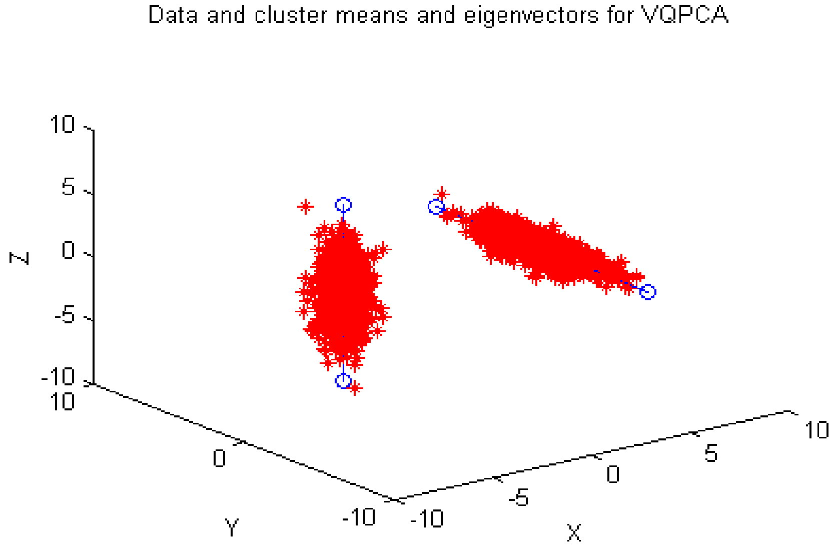

Figure 4.7. Cluster means and eigenvectors for VQPCA

As can be seen from Figure 4.7, the means and the eigenvectors correspond to the means and eigenvectors identified by the MPC. The corresponding squared for the VQPCA is shown in Figure 4.8.

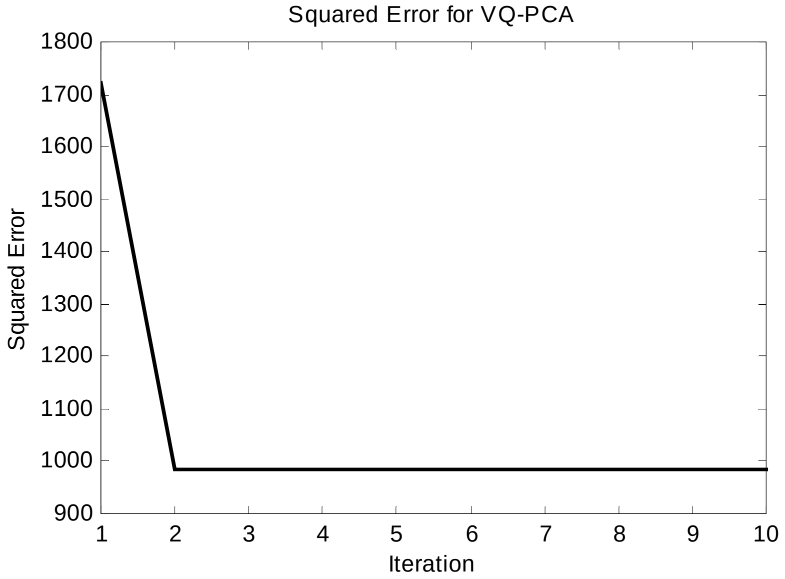

Figure 4.8. Squared error for the VQPCA

When we compare this squared error for the VQPCA with that for the MPC, we see that the squared error for the VQPCA is around twice as large as thereby showing that the MPC has better error performance. This is to be expected, as for the same set of means and eigenvectors, the VQPCA is a special case of the MPC, with the weights chosen so that one of the weights is one and the rest are all zero. The above discussion is just to illustrate that the error performance of the MPC is better than that of the VQPCA.

4.8. Error Concealment using Mixtures of Principal Components

Loss of data during transmission leads to objectionable visual distortion in the reconstructed video. Hence it is necessary for the decoder to perform error concealment, to post-process the video to remove these artifacts. Most of the existing work on error concealment uses spatial and temporal interpolation schemes.

We propose a model based error concealment scheme. We build models for regions of interest in the frame and use these models to replenish any missing data. Such a model based concealment approach is very useful especially for the MPEG-4 standard, which uses object based coding, thereby making it easy to determine regions of interest and build appropriate models for them. This model may be created from the data available or may be trained offline using some training data. Once the parameters of the model are estimated, this model may be used to represent the object and hence any missing data in the object in the frame may be replaced using data from the model. In order to capture the many variations among objects of interest we would like to model them using the MPC. For illustration, in our experiments we used faces as regions of interest, however the modeling technique is not restricted to any particular type of data.

As an illustration we show this error concealment scheme in Figure 4.9.

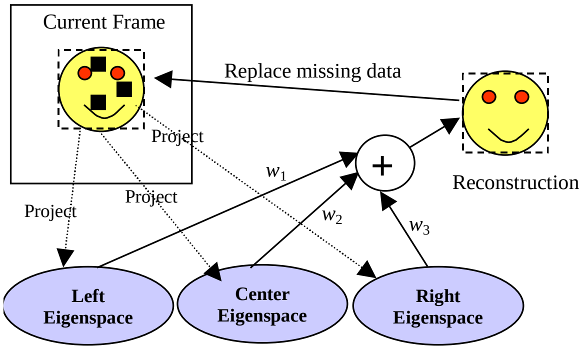

Figure 4.9. Mixture of eigenspaces for error concealment in faces, as illustration

In Figure 4.9, we name the three components of the eigenspace mixture, the left eigenspace, the right eigenspace and the center eigenspace to correspond to the natural poses of a human face. Each individual reconstruction is obtained in a way identical to the PCA reconstruction procedure. The final reconstruction of the data is obtained by linearly combining the individual reconstructions from each of these component eigenspaces, using a set of weights. This process of error concealment consists of two stages, first the projection onto the eigenspaces and linear combination of the component projections to obtain the final reconstruction and secondly replacing the missing data using the final reconstruction. Both these operations, of the projection onto the eigenspaces and the replacement of missing data may be viewed as projections onto convex sets. Each eigenspace itself is a convex set and a linear combination of data from many convex sets also leads to a convex set. So all the reconstructions lie in a convex set. Since the process of replacement only changes the missing data entries all vectors (the region of interest) in this set have the same known real values for the components not affected by error. Such a set is also convex. Given that both our operations are projections onto convex sets, we may iterate the process to obtain better results. This technique of POCS was used for image restoration by Sezan and Tekalp and more details on POCS may be found in [48]. Our error concealment scheme may be summarized as follows.

1. *Project region of interest onto the MPC to obtain reconstruction.*
2. *Replace missing data in region of interest using reconstruction.*
3. *Iterate 1 and 2 until convergence*

We collect a face sequence with the person moving his head from left to right, thereby showing broadly three poses, left, center and right. We cropped the faces in a 32×32 window using a face tracker developed by Huang and Chen [49] to provide the location of the faces. In order to illustrate the fact that the data actually comes from multiple clusters, we perform the following experiment. We determine the first three eigenvectors of the data using PCA and plot the corresponding coefficients in 3-D space. These are as shown in Figure 4.10.

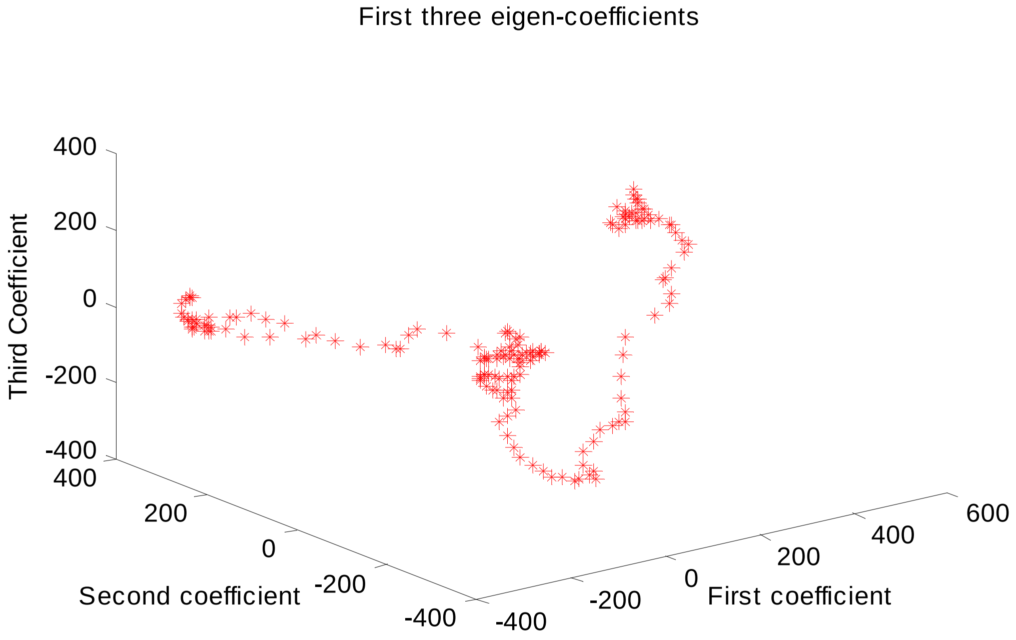

Figure 4.10. First three eigen-coefficients of real face data

From the plot we can see that the data may be approximated using three clusters, one each corresponding to the left, the right and the center poses. So we now try to represent the data using a MPC with three components, each one having two eigenvectors (making a total of six eigenvectors) and compare this with PCA with six eigenvectors. The squared error for the data representation using the MPC and using PCA is shown in Figure 4.11.

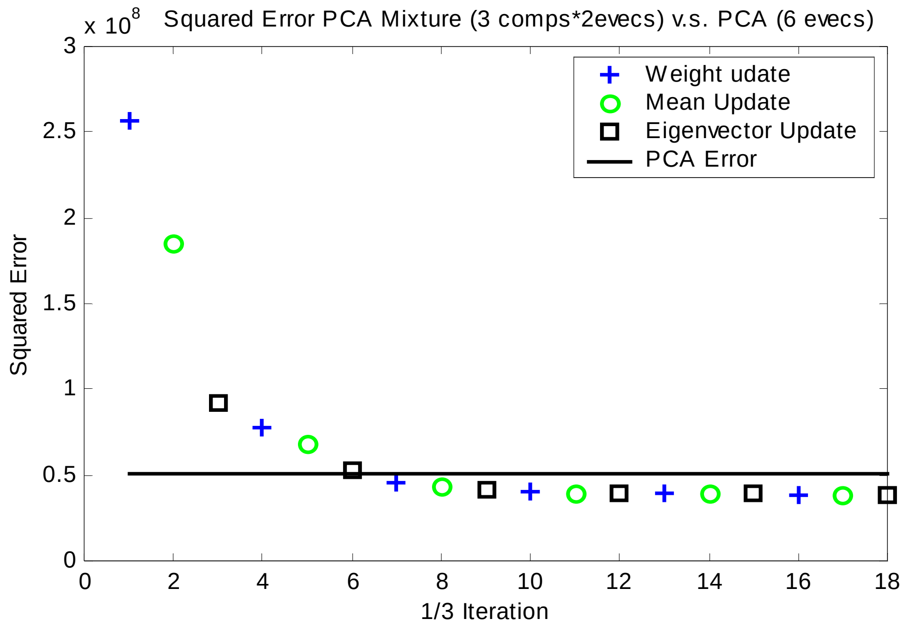

Figure 4.11. Squared error for face data

We can clearly see that using the MPC has smaller error, even with the same number of total eigenvectors as the PCA. In fact the error for the mixture is around 25% smaller than for the PCA. When we examine the means and eigenvectors for the resulting mixture components we see that they correspond to the left pose, the center pose and the right pose. These are shown in Figure 4.12.

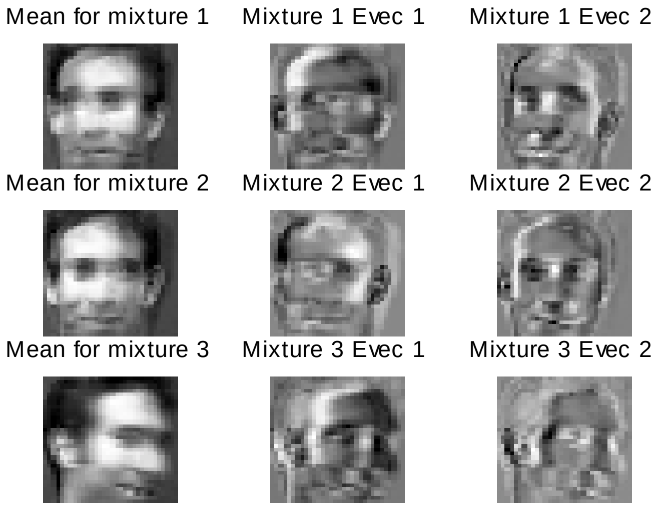

Figure 4.12. Means and eigenvectors for the MPC

We can see from Figure 4.12, that the means converge to the three poses, left, right and center. The eigenvectors also highlight the dominant motion associated with each cluster.

Now that we have a model to represent the data accurately, we use this model for error concealment. We first simulate bursty packet loss using a two-state Markov chain, a simple model proposed by Yajnik et al [50]. This model is shown in Figure 4.13.

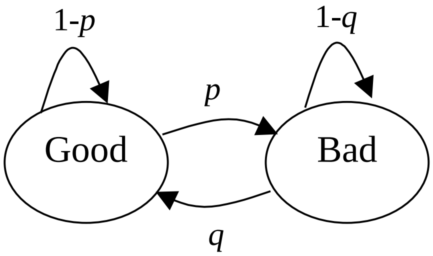

Figure 4.13. Two state Markov chain to simulate bursty packet loss

Each time we receive a packet we check the state of the Markov chain, if it is in the Bad state the packet is dropped and replaced by zero, otherwise the packet is successfully decoded. The parameters p and q of the Markov chain are selected as follows. We first choose q using the thumb rule that $(1-q)^{MBS} < T$, where the MBS is the maximum burst size, i.e., maximum number of consecutive bad packets and T is a very small number. This does not ensure that the MBS is not exceeded, however it is exceeded with a very small probability T. We then choose p so that the overall loss probability is as desired. p and q are related to the overall loss probability α as $\alpha = \frac{p}{p+q}$. For the purposes of our experiment we choose MBS to be 4 and vary the loss probability across different values. Also, for ease of implementation, in our experiments we use one packet to contain data for one 16×16 block. However, the scheme is not limited to such packet sizes, but may be applied across a variety of packet sizes.

The first experiment we perform is on Intra coded frames. For these frames, the loss of a block corresponds to the loss of actual image data and so missing blocks lead to a large drop in the quality. We test our concealment scheme on a sequence with 180 frames. We train the mixture and the PCA using 20 of the frames and we use these trained eigenspaces to conceal errors in the remaining 160 frames. In practice, the decoder is aware of which frames are clean and which frames have errors, so it is reasonable to assume that the model can be trained on some clean data before being used for error concealment. We perform two iterations of POCS to obtain better convergence results. As a measure of our error concealment performance we evaluate the PSNR between the frame with error concealment and the frame with no errors. We compare the error concealment performance using the mixture as well using the PCA. The PSNR values for these different concealment schemes are shown in Figure 4.14.

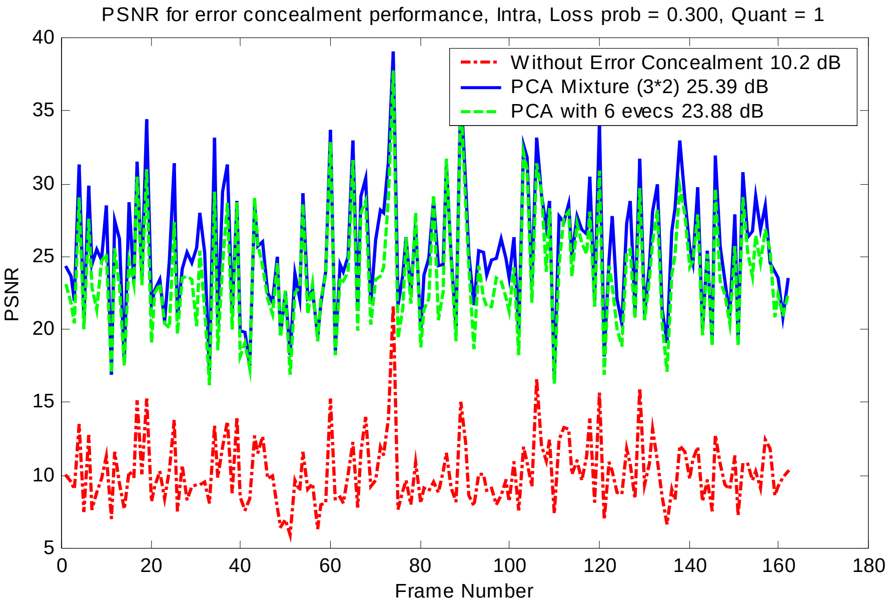

Figure 4.14. Error concealment results for Intra coded sequence

The results in Figure 4.14 are obtained for the case when the loss probability α is set to be 0.3 and a quantization step size of 1 is used while coding the sequence. From the figure we see that the MPC as a model for the data out-performs the PCA in terms of error concealment by around 1.5 dB. There is a 15 dB improvement over performing no error concealment and this is to be expected as the loss of any block leads to objectionable visual artifacts. We show some sample frames with error concealment from this sequence in Figure 4.15.

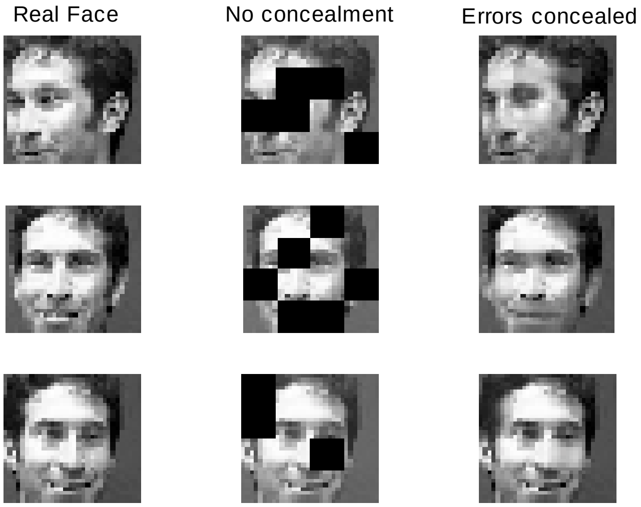

Figure 4.15. Error concealments using MPC for Intra case

In Figure 4.15 we show three sample frames with and without error concealment. The frames on the right have the errors concealed using the MPC as model. The no concealment case shows the missing blocks set to black, with the bursty nature of the errors indicated by the consecutive error blocks that occur on the faces.

We repeat the experiment across different loss probabilities and different quantization step sizes. We use varying quantization step sizes to illustrate the performance across different bit rates. Although a fixed quantization step size does not correspond to a fixed bit rate, we can use the quantization step size as indicative of the bit rate, with a large quantization step size corresponding to a low bit rate and a high quantization step size corresponding to a high bit rate. In each case the PSNR is computed with respect to the quantized sequence, however with no errors. These results are included for both the error concealment schemes in Figure 4.16.

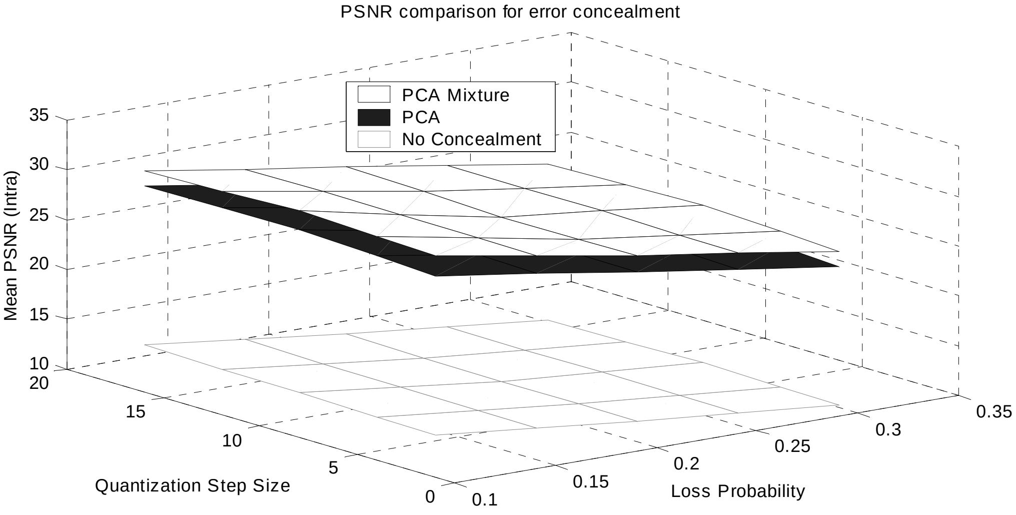

Figure 4.16. Error concealment across different loss probabilities and quantization step sizes (Intra)

We can see from Figure 4.16 that the MPC consistently outperform the error concealment using the PCA, even with the same number of total eigenvectors. On the average the improvement in error concealment performance is around 1.5~2 dB.

We then focus on error concealment for Inter coded sequences. For these sequences we assume that the motion vectors are available and packet loss corresponds to loss of the residue block. There are many advanced error resilient modes in both H.263 and MPEG-4 that allow for such availability of motion vectors under lossy conditions. Hence this means that we always have the motion compensated prediction for the current frame. The no concealment scenario involves replacing the residue block with zero, thereby using the motion compensated prediction from the previous frames. Using the motion compensated prediction is one of the error concealment schemes in practice currently. Although propagation of errors becomes a concern, still the visual artifacts are not as bad as when we lose data in the Intra case, especially when the sequence does not have too much arbitrary motion. As opposed to this, the MPC and PCA based concealment schemes project the face into the respective eigenspaces and replace blocks in the face for which the residue block was lost. We perform two iterations of the POCS to obtain these results. In order to highlight the improvements of using a model based concealment scheme as opposed to using the motion compensated prediction, we sub-sampled the sequence in time by a factor of three, thereby making the motion compensated prediction not as good as for the full frame-rate sequence. The results for the error concealment across the frames of the sequence are shown in Figure 4.17.

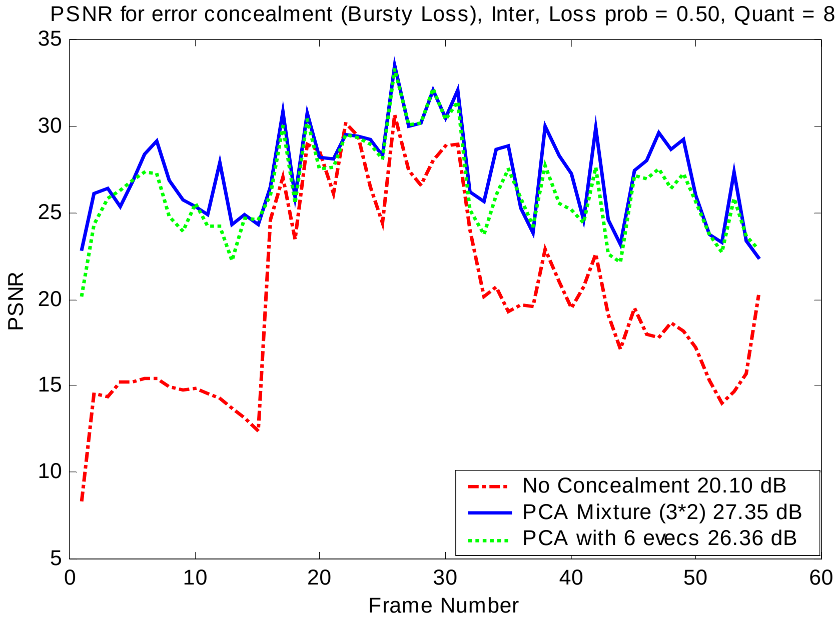

Figure 4.17. Error concealment Results for Inter coded frames

We can see from Figure 4.17 that the MPC consistently outperforms the PCA as a model for error concealment. The error concealment using the MPC has around 7. 25 dB better PSNR than the no concealment scheme and around 1 dB higher PSNR than the error concealment using PCA.

On the average the no concealment scheme has a PSNR of around 20.1 dB, which is much higher than the Intra no concealment case, since the no concealment scheme in the Inter case uses the motion compensated prediction. We notice that with no error concealment the PSNR degrades across time, as errors accumulate. However, there are frames when the no concealment PSNR increases, for instance we can see from the plot that around frame 18, the PSNR for the no concealment scheme rises sharply. This is explained by the fact that frame 18 has some intra coded blocks in the face region, and few errors, thereby leading to a virtual refresh. We show an example of three consecutive frames, frames 16, 17 and 18, in Figure 4.18 to highlight this.

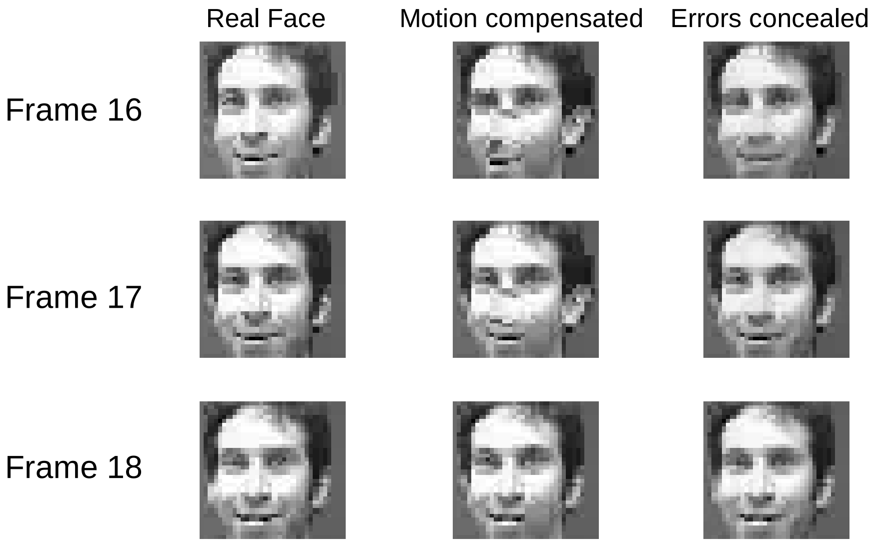

Figure 4.18. Sample frames with error concealment using MPC

We can see from Figure 4.18 that frames 16 and 17 not only have poor PSNR but the errors propagate between frame 16 and 17 due to the use of the motion compensated prediction, however due the presence of intra coded blocks and few errors in frame 18, the PSNR for the no concealment scheme improves. As against this, the frames with errors concealed using the MPC have a steadily high PSNR, although there is some smoothing due to the POCS iterations.

We then repeat the experiment across different loss rates and quantization step sizes and the results are shown in Figure 4.19.

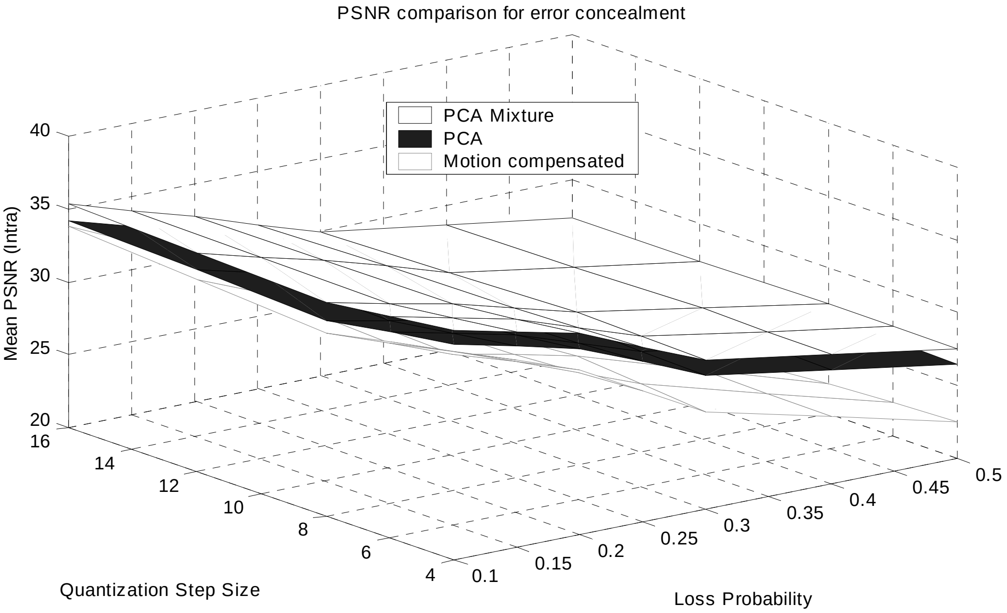

Figure 4.19. Error concealment across different loss probabilities and quantization step sizes (Inter)

We can see from Figure 4.19 that the error concealment using the MPC as a model performs consistently better than error concealment using the PCA, by around 1 dB across all the different quantization step sizes and loss probabilities. Both the PCA and the MPC based error concealment schemes work better than the motion compensated prediction used by the no concealment scheme. The MPC based error concealment outperforms the no concealment on the average by 5~7 dB. For the inter case we can tolerate loss probabilities of up to 0.5 due to the fact that we discard residue blocks instead of actual image data.

4.9. Mixture of Principal Components for Face Recognition

Face recognition has generated much interest in the research community primarily because of the multitude of applications it enables. Automatic face recognition is very useful as a non-intrusive authentication, verification and identification tool. Among face recognition techniques template matching techniques are very popular. Template matching involves building a template or model for each object in the database and then using that to classify the test face. A sample template matching recognition system is shown in Figure 4.20.

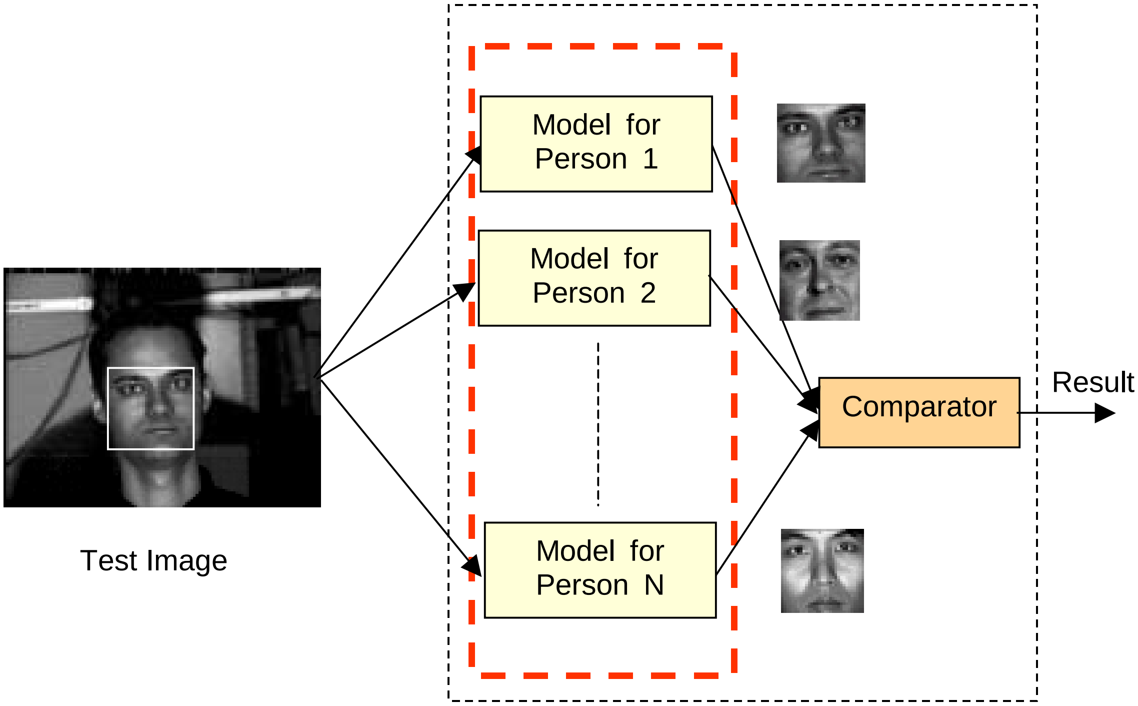

Figure 4.20. Sample face recognition system

In Figure 4.20 that the test face is matched with the models in the database, each of which returns a score (probability, likelihood, distance etc.) and all the scores are passed to a comparator that decides on the final result.

The eigenface approach for recognition was proposed by Turk and Pentland [51] who used PCA to create an eigenspace for all the subjects in the database. The test face is projected onto this eigenspace and the resulting coefficients are used to classify it among one of the many subjects. More recently other approaches have been proposed that use PCA to create individual eigenspaces [52], i.e., eigenspaces for each subject separately. The test face is projected onto these different eigenspaces and the eigenspace that has the smallest reconstruction error for the test face is chosen as the result.

As discussed earlier, the MPC is more efficient than the PCA at capturing data variations, especially when the data consists of multiple clusters. Face data with multiple poses and under different illuminations is likely to exhibit such multiple clusters, as illustrated by the simple example with pose variations in Figure 4.10. Hence, it is likely that using the MPC to represent each individual, instead of the PCA will lead to a better modeling performance. The consequence of having more accurate models for each individual is to improve recognition performance.

In order to verify these claims we use the MPC for a recognition task. The subjects we use are part of the Pose, Illumination and Expression (PIE) [53] database. This is a publicly available database collected at the Robotics Institute at Carnegie Mellon University. This database consists of 68 subjects showing 13 poses, under 23 different illumination conditions and with 4 different expressions, making a total of 41368 images. Of this database, we use five subjects with only pose and illumination variations as part of our recognition task. Sample images for these five subjects are shown in Figure 4.21.

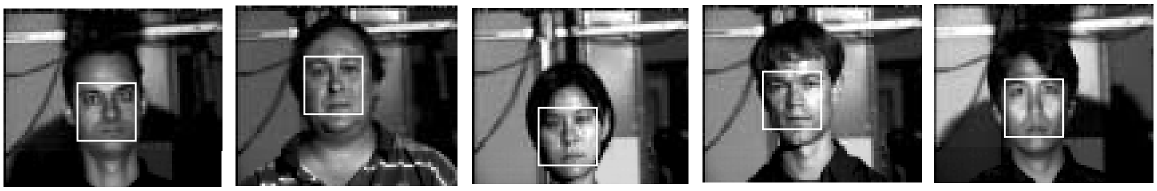

Figure 4.21. Sample images for five subjects in database

For each of the subjects shown in Figure 4.21, we use 13 poses with 22 illumination conditions, making a total of 286 images per person. Of these 286 images we use 143 for training and the rest for testing. The faces were cropped out from these images manually and then were resized to 32×32 to train the mixtures of eigenspaces. For each subject we train four mixture components with two eigenvectors each. We choose this number as it allows us to capture around 90% of the training data energy. Simultaneously we also train eigenspaces with 8 eigenvectors using PCA. An example of the mixture parameters at convergence for one of the subjects is shown in Figure 4.22.

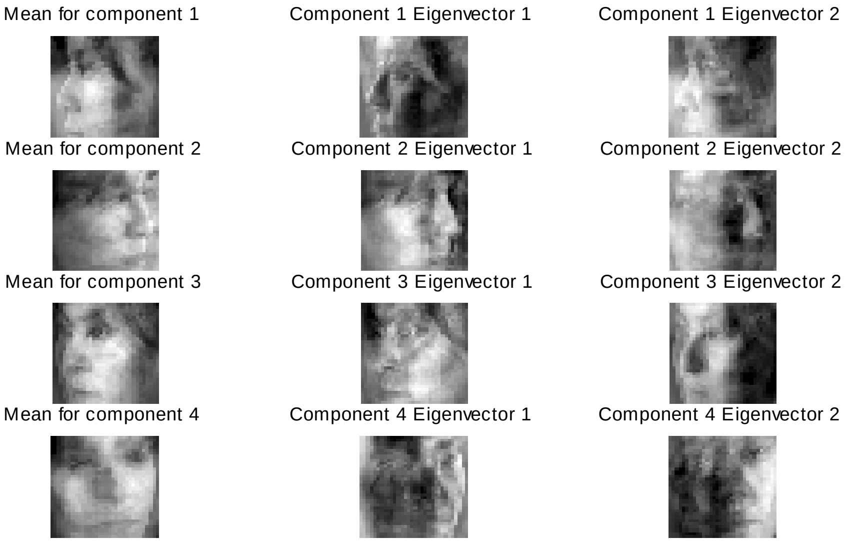
Mean for component 1
Component 1 Eigenvector 1
Component 1 Eigenvector 2
Mean for component 2
Component 2 Eigenvector 1
Component 2 Eigenvector 2
Mean for component 3
Component 3 Eigenvector 1
Component 3 Eigenvector 2
Mean for component 4
Component 4 Eigenvector 1
Component 4 Eigenvector 2

Figure 4.22. Converged parameters for MPC for Subject 3

As can be seen from Figure 4.22, the means for the four mixture components converge to different poses while the eigenvectors primarily capture the lighting variations within each component. This leads us to believe that the pose variations are more dominant than the lighting variations in this training set.

We then use these models for each subject in the recognition system shown in Figure 4.20 and evaluate the recognition performance. We project each test face onto the model for each person, evaluate the reconstruction error and assign the test face to the class with the smallest reconstruction error. Simultaneously we also collect recognition results using the PCA eigenspaces with 8 eigenvectors each to model each person. The results of the recognition are shown Figure 4.23.

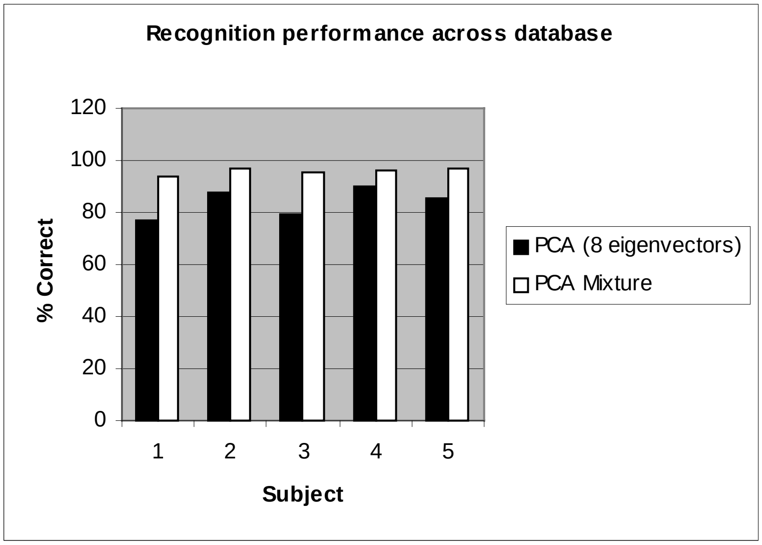

Figure 4.23. Recognition results for 5 subjects

From Figure 4.23 we can see that the recognition using MPC outperforms the PCA recognition across all the subjects in the database. The overall recognition using the MPC is 95.8% as opposed to 83.8% using the PCA, making sure that the total number of eigenvectors used for both cases is the same. We also highlight the confusion matrix for the MPC in Table 4-2.

Table 4-2. Confusion matrix for recognition with MPC

	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5
Subject 1	134	1	4	0	4
Subject 2	0	139	2	1	1
Subject 3	0	1	136	0	6
Subject 4	1	0	2	137	3
Subject 5	0	0	3	1	139

In Table 4-2, the different rows correspond to each subject, while the columns represent what the subject is recognized as. For instance, if we read off values from the first row we can see that Subject 1 is recognized as Subject 1, 134 times, as Subject 2, once, as Subject 3, four times, and as Subject 5, four times. As desired, the diagonal elements of this matrix are the largest values leading to good recognition performance. We show some examples of confusing faces in Figure 4.24.

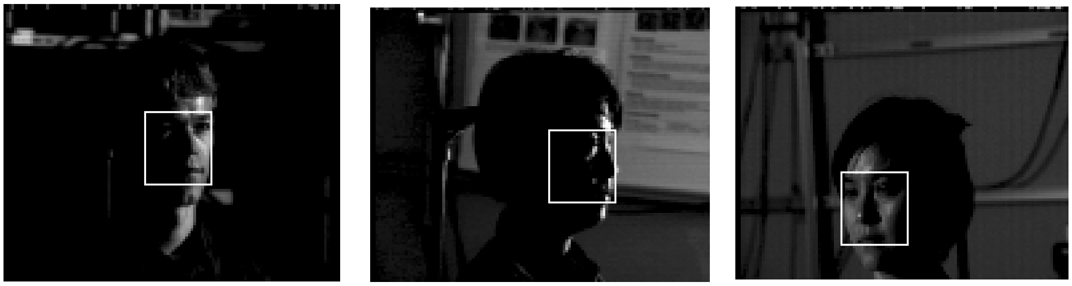

Figure 4.24. Confusing images in recognition database

The faces shown in Figure 4.24 are not recognized correctly by the system primarily due to the extreme dark lighting in the images, with a large portion of the face being dark. Overall, however the recognition system performance improves significantly when we replace the PCA with the MPC as a model for the subjects. The improvement for these five subjects is around 12% thereby supporting our claim that the MPC are indeed better models for data with large variations and this better modeling performance leads to the improved recognition performance.

4.10. Mixture of Principal Components for Tracking

In this section we describe a scheme to use the MPC for foreground background segmentation and illustrate how this may be used for face tracking. We collect data for a person in a moving car and the resulting sequence has extreme lighting, large changes in the face pose and rapid motion. These large variations make it difficult for conventional face trackers to track the face in this sequence. We show some sample frames from the sequence in Figure 4.25.

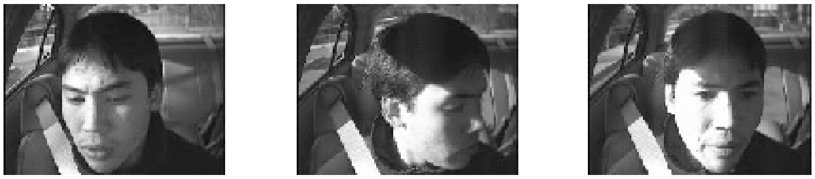

Figure 4.25. Sample frames from car sequence

We can see from Figure 4.25 that there are large variations in the face appearance caused due to the motion, pose changes and lighting variations. Conventional hue based trackers cannot account for these large changes and provide poor tracking performance. The MPC is very useful in capturing statistical properties of data with large variations. We may thus use it to model the background under these different lighting conditions so that we may segment out the foreground, and the foreground map may be then fed to a tracker to obtain a reliable result. This scheme is shown in Figure 4.26.

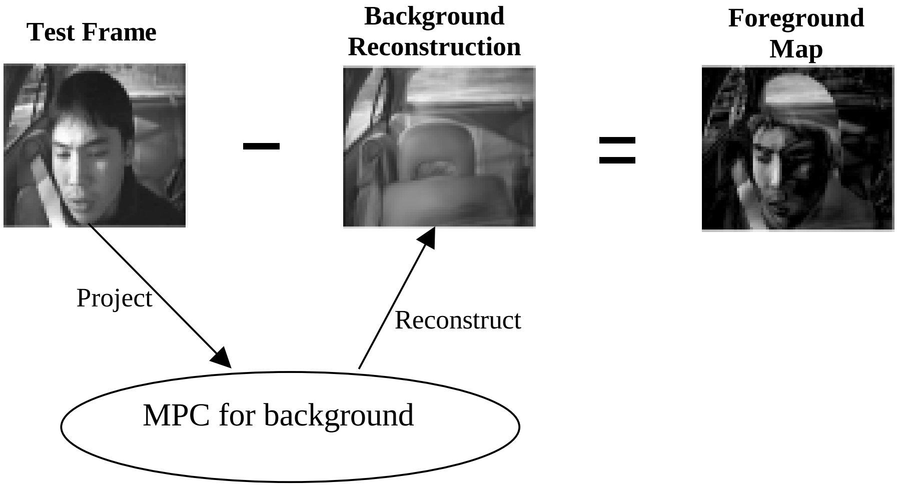

Figure 4.26. Foreground-background segmentation using MPC for background

In Figure 4.26 we build a model for the background under different lighting conditions. In order to train this model we collect a sequence in the moving car, but without the person in the seat. This provides us with the background under different lighting variations. This model may be then used to reconstruct the background in any test frame hence we may obtain the foreground map, which is used by the tracker to provide a stable tracking result. We model the background using a mixture of six eigenspaces, with two eigenvectors each. The resulting means for the six components are shown in Figure 4.27.

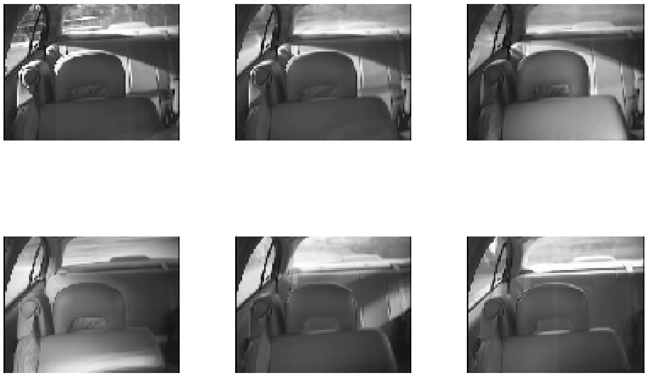

Figure 4.27. Mean frames for the six components

In Figure 4.27 we show the six means for the different components of our model. Clearly, these correspond to different lighting conditions. We evaluated the performance of the proposed scheme on the real video sequence and preliminary results indicate that the tracking result is stable and can maintain track even across rapid pose and lighting changes. We show some results of the tracking in Figure 4.28.

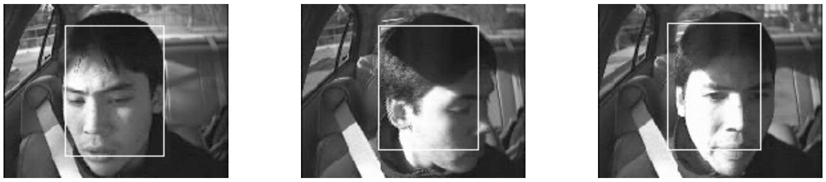

Figure 4.28. Sample tracking results

As may be seen, the results of the tracking, shown as the white rectangular box in the figure, are stable across different lighting conditions as well as across different poses of the user. Of course since the tracking is based on the segmentation result, the hair, which is an important distinguishing factor for segmentation, is also considered part of the face. This may be corrected by fusing the tracking results with a color or hue based tracker.

4.11. Conclusion

We have introduced MPC to model data that belongs to multiple clusters and have described an EM like algorithm to train the parameters for the MPC. We have shown using simulated data that using the MPC provides us much better performance than using the PCA, even when both use the same number of parameters. We then use the mixtures of PCA for error concealment of video sequences with losses due to transmission over networks. We perform concealment for both Intra coded as well as Inter coded sequences across a variety of quantization step sizes and packet loss probabilities. We show that using a model based approach to error concealment leads to very good error concealment performance, measured by examining the PSNR as compared to the error free video sequence. We show that using the MPC as a model leads to better error concealment performance than using the PCA as a model by around 1~2 dB, even with the same number of total parameters. We also show that model based error concealment provides much better performance by around 5~7 dB over the traditional use of motion compensated residue for concealment. We then use the MPC to improve a face recognition system. We use the MPC to model lighting and pose variations for each subject in the test database and realize that the means of the components converge to the different poses while the eigenvectors for each component capture the lighting variations. We train four mixture components with two eigenvectors for each subject and simultaneously also train PCA with 8 eigenvectors for each subject. We test recognition over five subjects and show that using the MPC leads to an improved recognition performance by around 12%. Thus the MPC may be used to capture pose and lighting variations efficiently leading to an improved recognition performance over the PCA. We also illustrate the use of the MPC in foreground background segmentation and the use of the segmentation result to obtain robust tracking performance. It is very important to realize that the MPC may be used to model any kind of data with large amounts of variation thereby making is suitable for any task requiring statistical modeling tools and across any domain, not just the pixel based spatial domain.

5. Modeling of Variable Bit Rate Video Traffic

This chapter describes the statistical models we develop to capture the variations in real video traffic. These models are not actually part of the encoding or decoding process, however they play a significant role in video coding optimization. Accurate models for video traffic may be used to determine the network state and this information is very useful to network providers and designers as it allows them to evaluate the performance of the network in order to make certain guarantees and predictions of the network behavior. This information may also be used by the encoder and the decoder to improve the coding performance.

Variable bit rate (VBR) video coding allows for great flexibility in terms of selection of video coding parameters, efficient compression ratios and can maintain desired video quality. Bitstreams from various VBR video sources can also be efficiently multiplexed over the network using statistical multiplexing techniques. All the above factors have led to VBR video encoders being the preferred mode of coding video streams and the focus of this paper is on modeling such video sources. Modeling video sources is important as it allows for network and video codec designers to estimate the parameters of networks like packet loss probabilities and end-to-end delays so that they can guarantee a desired quality of service (QoS).

This chapter introduces flexible models for VBR traffic that allow for the various characteristics of such traffic, such as variable GOP structure, different activity levels and different frame types. We use doubly Markov models and AR processes to capture these properties. Most of the prior work in modeling VBR traces assumes that modeling the mean and the variance of the data is sufficient to capture the stochastic properties of the trace. So the modeled traces are created using a Gaussian probability density function (pdf) for the noise, with the appropriate means and variances. We show, by examining the real pdf of the data traces, that this Gaussian density assumption for the model is not accurate. Instead, real data has pdfs that are better modeled using Exponential distributions and this Gaussian assumption, hence limits the performance of the models.

This chapter is organized as follows. Section 5.1 describes the models and Section 5.1.3.2 includes a discussion of results in terms of statistical parameters as well as in terms of network simulations. Section 5.2 includes the extension of the models to capture the probability density of the data accurately, and some simulation results with the modified model. We then conclude with the summary of the models and their performance.

5.1. Activity Adaptive Models

Video sequences have large variations in action levels between scenes. This leads to large variations in the bits per frame within I, P or B frames, corresponding to different activity levels. An accurate model needs to capture the effect of having different frame types, as well as this variation in activity level within frames of one type. We thus propose a number of doubly stochastic processes to model both the activity level changes and I, P and B frames corresponding to a certain activity level. As before, the temporal correlation between I, P or B frames corresponding to an activity level is captured using AR(1) processes with Gaussian distributions. Our models are flexible and allow for a variable GOP structure.

5.1.1. Trace Characteristics

Typical traces with I, P and B frames and variable GOP structure are created as follows. The video encoder first identifies which frames of the sequence need to be coded as I frames. These are frames that lie across scene changes and so cannot be coded efficiently using prediction. This may be determined by creating a prediction for every frame and counting the number of blocks in the frame that need to be intra coded, i.e., cannot be predicted well. Frames that have a large number of intra coded blocks are classified as I frames. After identifying the I frames the sequence is encoded with a repeating the pattern of two B frames followed by a P frame, until the next I frame is reached. Clearly, when the interval between two I frames is not a multiple of three this pattern cannot always be inserted. In that case the last pattern is terminated when the I frame is reached. An example sequence of coded frames is as shown in Figure 5.1.

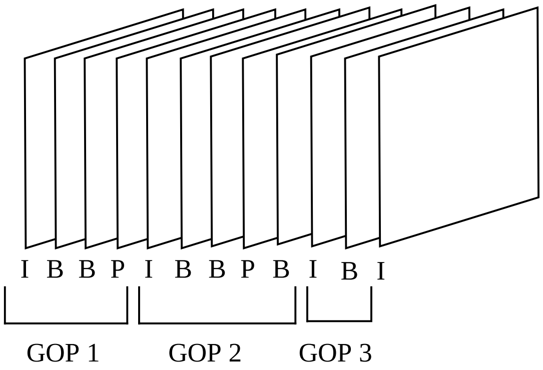

Figure 5.1. Variable Length GOPs with B frames

From the sequence we see that the interval between the first two I frames is a multiple of three and so the BBP pattern can be repeated once, but after this none of the successive I frames have an interval that is a multiple of three, so in all the other cases the BBP pattern is terminated when the next I frame is reached. Traces generated in such a way have a variable length GOP structure as well as all three kinds of frames. We propose a number of doubly stochastic processes to model both the activity level changes and I, P and B frames corresponding to a certain activity level. The temporal correlation between I, P or B frames corresponding to an activity level is captured using AR(1) processes with Gaussian distributions.

5.1.2. Type I Models

We divide the video sequence frames into three different activity levels, high-activity, medium-activity and low-activity, based on the number of bits needed to code the frame. The Type I models, as for the I and P models, choose between generating an I frame, a P frame or a B frame before deciding the activity level of the video frame. The traces that we wish to model have some specific characteristics. For instance there are a lot of repeated pairs of B frames, but no instance of three or more consecutive B frames. Similarly, P frames never occur consecutively due to the way in which we create the traces that we wish to model. So we may modify the structure of the Markov chain corresponding to I, P and B frames in order to account for these specifics in our traces. Hence, instead of having only three states corresponding to I, P and B frames, we introduce an artificial fourth state that we call the BB state. For each transition into this state two B frames are produced. As against this transitions into any of the other three states, I, P or B, result in only one frame being generated. This BB state simulates the repeated B frame structure in our real traces. The other constraints on having no more than two repeated B frames and having no repeated P frames are automatically satisfied when we estimate the transition probabilities of the model from our training data.

5.1.2.1. Type I Doubly Markov Model

This model has two Markov chains, the outer one having I, P, B and BB states, as described earlier, and the inner one having states corresponding to the activity levels of the type of frame generated. Each of the frames generated, may belong to one of the three activity levels. Our proposed model looks as shown in Figure 5.2.

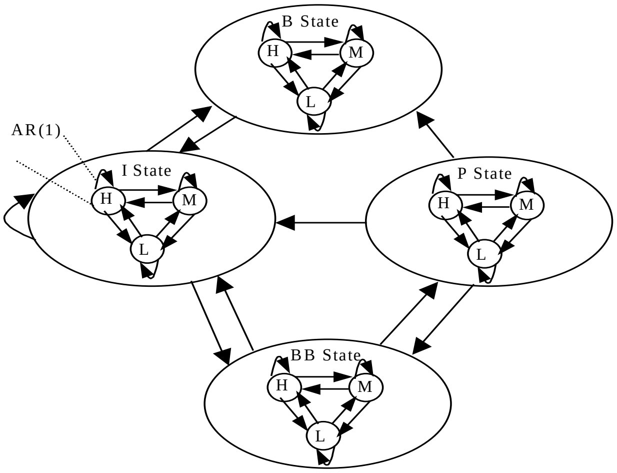

Figure 5.2. Type I Double Markov Model for trace with B frames

As can be seen from the figure, the outer Markov chain has some constraints on its structure due to the particular characteristics of the data we are trying to model. For instance, there are no transitions in either direction between the BB state and the B state, or self transitions for either of these states, to prevent the occurrence of three or more consecutive B frames. Similarly there are no self-transitions in the P state, as we cannot have consecutive P frames. Also, we realize that we generate a single B frame only when the BBP pattern needs to be prematurely terminated, so the B state may transition only to the I state. All these conditions need not be imposed on the model, training it using the real trace will ensure this transition, structure, however we may use this apriori knowledge to reduce the number of parameters to be estimated during training.

Within each of the four outer states, we have another Markov chain that determines the activity level of the frame to be generated. These activity Markov chains are never restarted. The process of generating data using this model is as follows. We first decide to generate an I frame, a P frame, a B frame or two consecutive B frames. After this, we determine the activity level of the frames using the inner Markov chain. The frames are generated from a Gaussian probability density function (pdf), using an AR(1) processes to capture the temporal correlation between them. The two frames in the BB state are generated from the same activity level.

The training procedure for this model is as follows. Using the sequence of I, P, B and repeated B frames from the real trace we may estimate the initial and transition probabilities of the outer Markov chain. We then need to estimate the inner Markov chain initial and transition probabilities and the means, variances and AR process parameter ρ for each of the AR(1) processes. We separately collect all I frames, all P frames, all single B frames and all repeated B frames from the real trace. We then use two thresholds for each of these and divide them into high-activity, medium-activity and low-activity. From these sets of data we can estimate the means, variances and AR process parameter ρ of all the AR(1) processes. By looking at the sequence of transitions between these activity levels for each of the four frame types, I, P, B or BB, we can estimate the initial and transition probabilities for the inner Markov chain.

5.1.2.2. Type I Simplified Model

As before, we try to reduce the parameters for the model by removing Markov chains when they are not necessary. For the I and B states we found experimentally that $P(S(n)=S_i \mid S(n-1)=S_j) \approx P(S(n)=S_i)$ for the inner Markov chain. This means that the transition probabilities between the inner Markov chain states are the same as the unconditional probabilities of being in any of them, so we can replace the Markov chain with a set of unconditional probabilities with which we generate a frame belonging to a certain activity level. This may be explained by the fact that the I and B states occur infrequently and always in different GOPs, typically with a large interval between them. So the dependence of the current activity level on the previous activity level is small. This is however not true for the BB or the P states as they occur frequently and many times in the same GOP. Hence these Markov chains cannot be replaced. The simplified model is as shown in Figure 5.3.

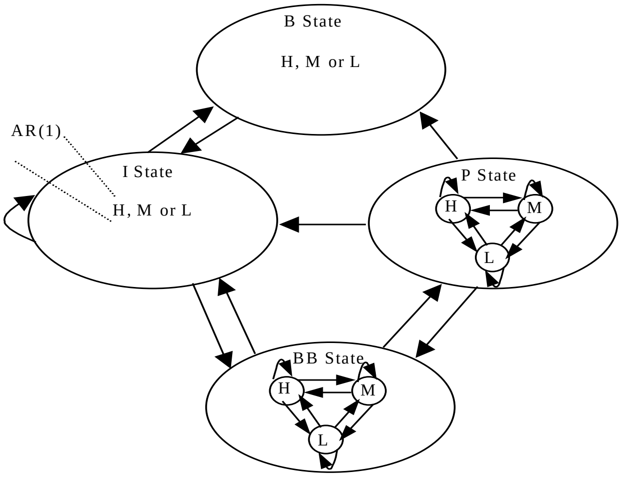

Figure 5.3. Type I Simplified Model

As for the Doubly Markov model, the structure of the model is chosen keeping in mind the particular characteristics of the data we are trying to model. For this model, we first decide whether we want to generate an I frame, a P frame, an individual B frame or a pair of B frames and following this we decide which activity level this frame/frames should belong to. For the I and B states we decide with a fixed probability the activity level of the generated frame, while for the BB and P states we use the Markov chain to determine the activity level of the frames. As before, none of the inner activity Markov chains are restarted.

The training procedure for this model is very similar to that for the Doubly Markov model. The only difference is that for the I and the B states the unconditional probability of generating a frame belonging to a certain activity level is just the number of frames at that activity level divided by the total number of frames of that type.

5.1.3. Type II Models

We also implement the Type II models where we first choose which activity state the current frame belongs to before deciding to generate an I, P or a B frame. Here again, using knowledge of the training data characteristics, we choose to use the four state I, P, B and BB model to generate frames. This Markov chain, is now, however the inner Markov chain for this model.

5.1.3.1. Type II Doubly Markov Model

We also implement the Type II model, where the outer Markov chain corresponds to the activity level and within each activity state there is another Markov chain corresponding to the I, P, B and BB states. This model may be shown as in the following figure.

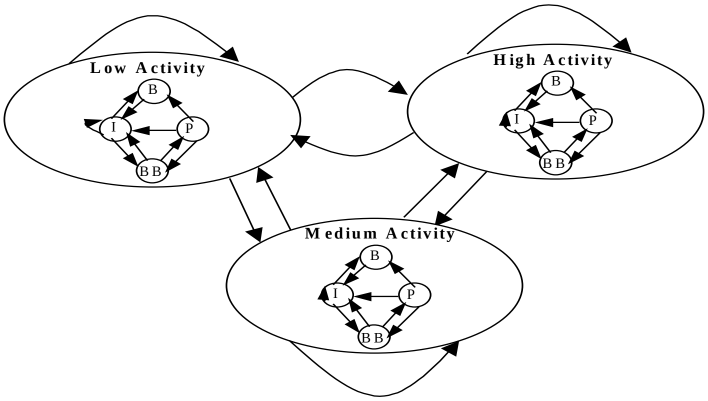

Figure 5.4. Type II Doubly Markov Model

This model makes the assumption that the entire GOP belongs to one activity level, however, this assumption is not very valid and the performance of the model suffers due to this. In order to generate data using this model, we first use the outer Markov chain to decide the activity level of the current GOP, following which we generate the GOP using the inner Markov chain. When the inner Markov chain transitions back to the I state, we have completed generating one GOP, and so we use the outer Markov chain to determine the activity level of the next GOP. We reinitialize all inner Markov chains after generating one GOP.

In order to train this model, we first obtain the mean bit rate for each GOP and then use two empirical thresholds to partition this sequence of GOPs into activity levels. This sequence of activity levels may be used to train the outer Markov chain. The means, variances and AR process parameter ρ for each of the AR(1) processes may be estimated using the knowledge of the activity level of each frame. We then collect sets of sequences of I, P, B and repeated B frames for each activity level and each of these sets is used to train the initial and transition probabilities for the inner Markov chains.

Our training data shows a great dependency in terms of deciding the current frame based on the previous frame and so we cannot replace any of the inner Markov chains with unconditional probabilities. So, this model cannot be simplified further.

5.1.3.2. Results and Discussion

In order to evaluate the performance of our models we look at both the stochastic properties of the generated data as well as use network simulations to look at the loss probabilities and delays encountered by the traces. All the models were trained on the same data and characteristics of the generated bit rate were compared with those of the real data. The training data was from two different sequences. The first was a high motion video sequence made up of advertisements. We call this sequence Ads. This sequence had frequent scene changes, camera zooms and pans and a lot of motion. The second sequence was a news clip and we call it News. This sequence contained news reports from different locations and hence it contained a moderate amount of motion and some scene changes. Sample frames from both the sequences are shown in Figure 5.5.

Figure 5.5. Sample frames from Ads (left) and News

Both sequences consisted of five minutes of data sampled at 15 Hz, making a total of 4500 frames. Each sequence was converted to bits using a H.263 standard compliant video codec. A random GOP was achieved as described in Section 5.1.1, and a frame is coded as an I frame if more than 70% of its blocks need to be intra coded. In order to illustrate the need for the flexible GOP structure, we compare our results with the Fixed GOP model.

5.1.4. Stochastic Properties of Modeled Traces

We compare the mean squared error in modeling the real autocorrelation function by our models with the error using the trace generated by the Fixed GOP model. The mean squared error in autocorrelation function using our proposed models, is smaller by an order of magnitude for both the Ads as well as the News sequence. These results are included in Table 5-1 with the entry in each column corresponding to the mean squared error in modeling real autocorrelation function normalized by the error in modeling the real autocorrelation function using the Fixed GOP model.

Table 5-1. Error in modeling real autocorrelation function normalised by Fixed GOP error

Sequence	Fixed GOP Model Error	Type I Doubly Markov Error	Type I Simplified Error	Type II Doubly Markov Error
Ads	1	0.083	0.081	0.101
News	1	0.074	0.077	0.092

From the table we can see that our models produce traces that are statistically similar to the real data as the error is around 10~13 times smaller than that for a fixed GOP model. From the table we can see that our models produce traces that are statistically similar to the real data. As an example of the traces generated we show 3000 samples of the trace and the autocorrelation function for the real data and traces from models in Figure 5.6, Figure 5.7 and Figure 5.8.

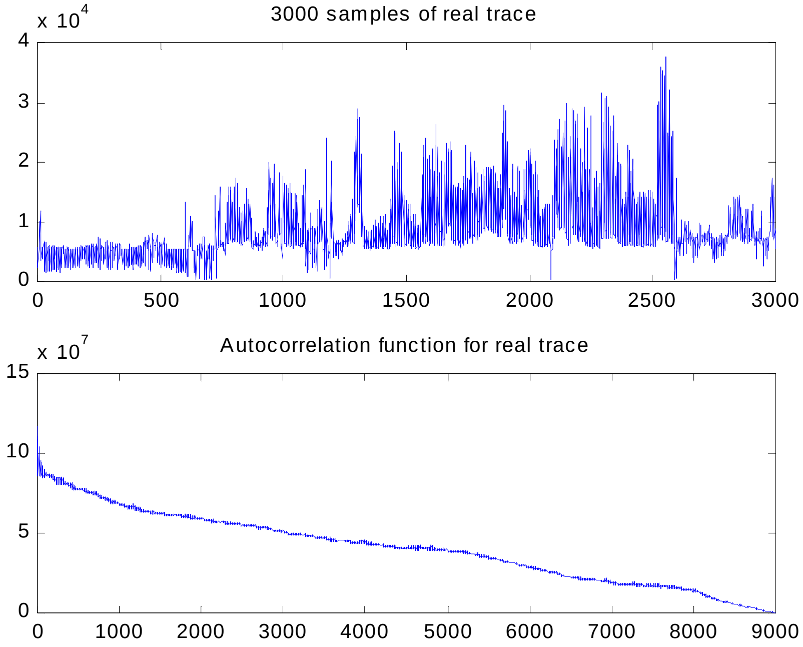

Figure 5.6. Real trace and its autocorrelation function.

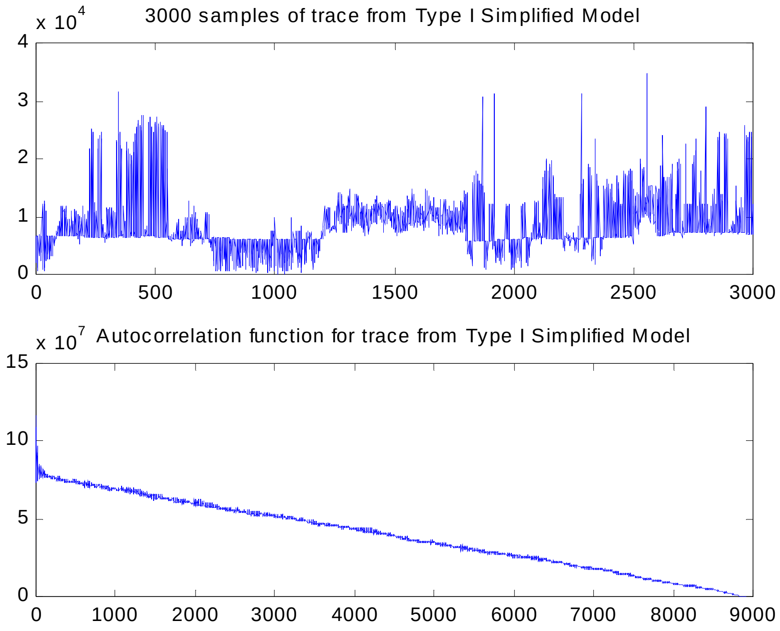

Figure 5.7. Trace and autocorrelation function for simulated data

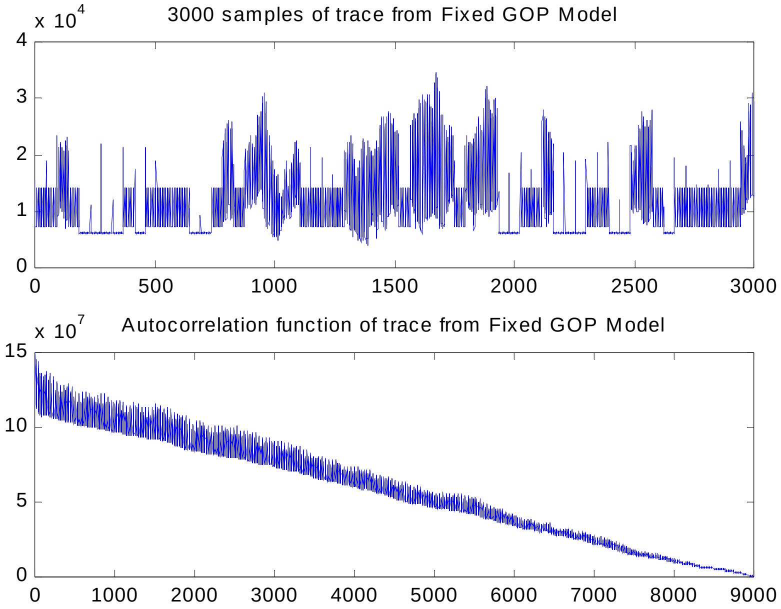

Figure 5.8. Trace and autocorrelation function for data from Fixed GOP model

From the figures we see that the data generated by our proposed models produces a trace that is similar to the real trace. This may be seen by the similarity in the autocorrelation function and the fact that the trace clearly exhibits different activity levels and a variable GOP structure, measured as the length between two I frames. As opposed to this the fixed GOP model is constrained by having to use a fixed GOP structure. This means that I frames occur at regular intervals and hence the trace that exhibits periodicity. This periodicity is evident when we examine the autocorrelation function and manifests itself in the regularly occurring spikes

5.1.5. Network Simulations

In order to actually translate this statistical similarity into the ability to actually predict the packet loss probabilities and delay, we perform the following simulation. We first look at a set of real traces and packetize them with one frame being viewed as one packet. We view each trace as being generated by a different video source and packets from these different sources are statistically multiplexed into a common buffer. Since each packet corresponds to a frame, our packets arrive at regular intervals, thereby leading to a certain periodicity in the system. In order to reduce this periodicity we uniformly distribute the starting times of the different sources within one frame interval of each other. So packets from the same source arrive at regular intervals of one another, but the packets from different sources start arriving at different times, which are uniformly distributed within one frame interval. All these packets enter a buffer with a fixed size, which is then drained at a fixed drain rate. The setup for this simulation is as shown in Figure 5.9.

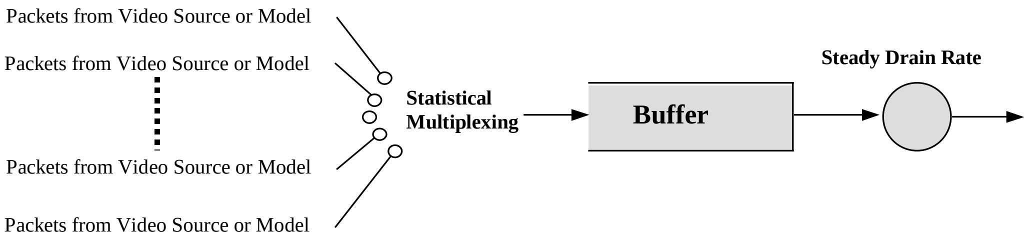
Packets from Video Source or Model
Packets from Video Source or Model
Packets from Video Source or Model
Packets from Video Source or Model
Statistical
Multiplexing
Buffer
Steady Drain Rate

Figure 5.9. Simulation setup to evaluate delay and loss probability for statistically multiplexed trace

As shown in Figure 5.9 packets from multiple sources are multiplexed into the buffer using statistical multiplexing, i.e., a first in first out (FIFO) scheme without any resource reservation. We evaluate the loss probability and delay encountered by the packets in this set up, and repeat this experiment for different buffer sizes and drain rates. A packet is considered lost when on its arrival the buffer is not sufficiently empty. Delay is measured for transmitted packets and includes both the waiting time in the buffer as well as the transmission time (packet size divided by the buffer drain rate).

Using the same setup we then replace each video source by our model and evaluate the loss probabilities and delay for our models. For comparison, we also replace each source with the fixed GOP model and evaluate the loss probability and delay for the packets generated by this model. We perform this experiment using traces from the Ads sequence and repeat it for traces from the News sequence.

In our simulation we use five source or model traces. We perform the experiment twice, the first time when all the sources are operating at the same quantization step size and the second time when some sources operate with different quantization step sizes. When the traces are generated using the same quantization step size their bit rates are comparable, whereas when the traces are generated using different quantization step sizes, their bit rates show a large variation. These two experiments are performed to examine the properties of the trace under different scenarios.

The results of the first simulation are shown in the following figures. Since we have all sources at one quantization step size, one model may be used to generate all the multiplexed traces for one kind of sequence (Ads or News). We present these simulation results for the trace generated by one of our models, the Type I Simplified Model and the trace generated using the Fixed GOP model in Figure 5.10 and Figure 5.11.

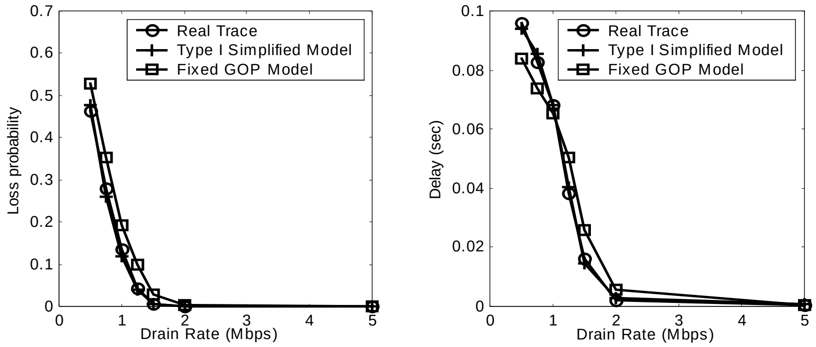

Figure 5.10. Loss probability and delay for real and modeled traces (Ads) using multiplexed streams

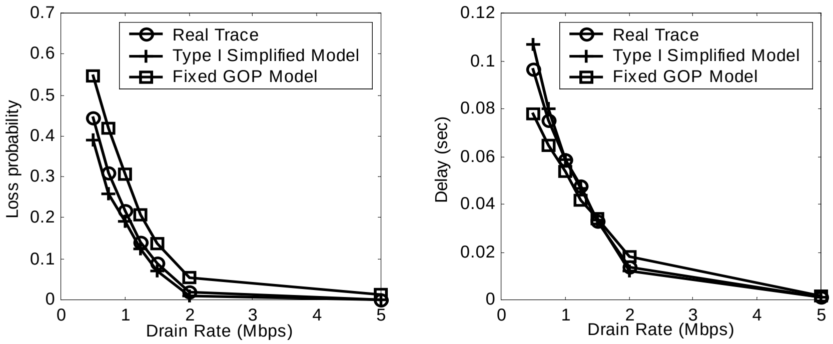

Figure 5.11. Loss probability and delay for real and modeled traces (News) using multiplexed streams

Figure 5.10 shows loss probability and delay results for statistically multiplexed streams for the Ads sequence, while Figure 5.11 shows results for the News sequence. We can see from the figures that the performance of our model is better than the performance of the fixed GOP model. Our predictions for loss probability and delay are within 4~15% of the actual values. In terms of squared error, our prediction of the loss probability for the real data is 18 times smaller for Ads and 6 times smaller for News than the error for the fixed GOP model. Similarly the squared error in our prediction of the delay is around 20 times smaller for Ads and 3.5 times smaller for News than the error for the fixed GOP model. The gains for the Ads sequence are larger as it is a high motion sequence with frequent scene changes and changes in activity levels, so a fixed GOP model cannot accurately capture all these variations.

The second simulation involves multiplexing traces from sources using different quantization step sizes. This is more representative of the real scenario when we have many traffic flows with different parameters. For our simulation we multiplex two sources using a quantization step size 8, two using a quantization step size 16 and one using a quantization step size 24. We need to train different models for these different traces, as the quantization step size variation is too large to be captured using one model. As before, we examine the loss probability and delay predicted by our model and the fixed GOP model as compared to the actual delay and loss probability. For the sources with different parameters our models predict delay and loss probability within 7% of the actual values. As compared to the fixed GOP model the squared error in prediction of loss probability is around 30 times smaller for the Ads sequence and around 12 times smaller for the News sequence. Similarly the prediction in delay is also around 35 times smaller for the Ads sequence and around 15 times smaller for the News sequence. Our models may hence be used to predict accurately the delay and loss probabilities encountered by multiplexing traces into a common buffer, whether we have traces with the same parameters or traces with different parameters.

5.2. Probability Density Function for AR processes

Autoregressive (AR) processes are wide sense stationary stochastic processes that model certain temporal relationships between consecutive samples. An AR(1) process is an AR process of order one. This means that there is some dependence of every sample on the previous sample. An AR(1) process $X(n)$ may be represented as in equation (5.1).

$$X(n) = \rho X(n-1) + W(n) \text{ with } |\rho| \leq 1 \tag{5.1}$$

In equation (5.1), $W(n)$ is white noise and ρ is the AR process parameter that defines the extent of dependence between samples. This corresponds to a special temporal relationship between samples, which is reflected in the autocorrelation function for the process. For such an AR(1) process, the autocorrelation function $R_{xx}(k) = E[X(n)X(n-k)] = \sigma^2 \rho^{|k|}$ is an exponentially decaying function, when the variance of $W(n)$, σ_W^2 is related to the AR process variance, σ^2 as $\sigma_W^2 = (1-\rho^2)\sigma^2$. There are many real life processes that may be modeled by an AR process. In particular when we examine traces generated by VBR sources we see that the temporal relationship between consecutive samples of the same frame type is well captured by an AR(1) process. All of the above discussion assumes that the stochastic process $X(n)$ is zero mean. However this is not true for our data since the data consists of actual bits per frame, which cannot be negative. Hence we first remove the mean from the data and then model the autocorrelation function of the mean-removed data using an AR(1) process. As an illustration of the fact that we may model the data using AR(1) processes, we do the following. We first collect all I frames from the Ads sequence that belong to the medium activity level. We then compute the autocorrelation function for these frames after removing the mean and show this function in Figure 5.12. In the figure, we also superimpose on this plot an exponentially decaying function to show that the AR(1) model is a good model for real data.

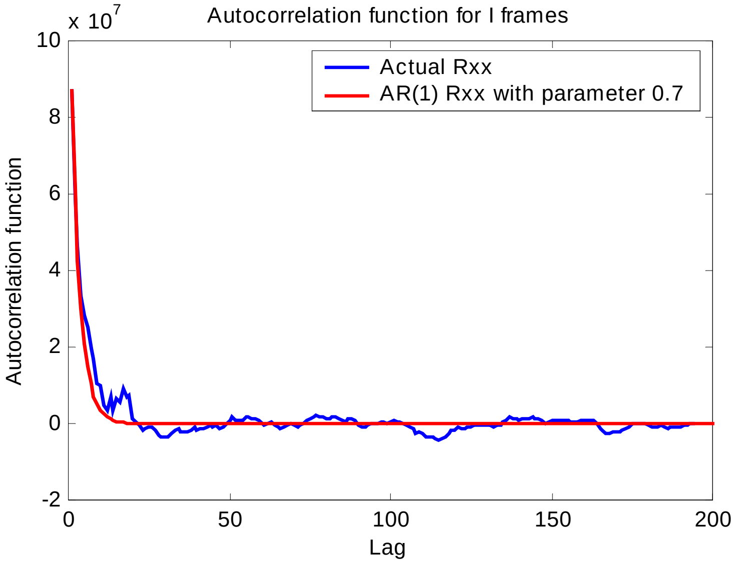

Figure 5.12. Modeling of real autocorrelation function (mean-removed) using AR(1) process

In Figure 5.12, we show the autocorrelation function for I frames belonging to the medium activity level and show that an AR(1) model with parameter $\rho = 0.7$ can capture the temporal correlation quite accurately. However, it is not enough just to capture the temporal correlations, since each sample of the process has a certain probability density function (pdf) associated with it. We need to capture this information also. Most previous work on modeling VBR video traces does not focus on this aspect and simply models the distribution using a

Gaussian with the mean and variance corresponding to the data. This approach is often inaccurate, as the pdf of the data may not conform to the Gaussian assumption.

If $X(n-1)$ has a pdf $f_{X(n-1)}(x)$, then $\rho X(n-1)$ has a pdf $\frac{1}{|\rho|} f_{X(n-1)}\left(\frac{x}{\rho}\right)$ due to the scaling factor ρ. We know that $W(n)$ and $\rho X(n-1)$ are uncorrelated, since $\rho X(n-1)$ depends only on past values of the noise and the noise is white, i.e., noise samples are uncorrelated. Hence when we sum $W(n)$ and $\rho X(n-1)$, the resulting pdf may be obtained as a convolution of their individual pdfs. So, if $W(n)$ has a pdf $g_W(x)$ then the pdf of $X(n)$, $f_{X(n)}(x)$ is

$$f_{X(n)}(x) = \frac{1}{|\rho|} f_{X(n-1)}\left(\frac{x}{\rho}\right) * g_W(x) \tag{5.2}$$

where $*$ corresponds to convolution between the two functions. If samples of the AR process are to have the same pdf $f(x)$ we need $f_{X(n)}(x)$ to be the same as $f_{X(n-1)}(x)$, or

$$f(x) = \frac{1}{|\rho|} f\left(\frac{x}{\rho}\right) * g_W(x) \tag{5.3}$$

Equation (5.3) provides us a relationship between the noise pdf and the desired pdf for the AR(1) process. Hence for the data samples to have a specified pdf $f(x)$, we need to solve equation (5.3) for $g_W(x)$. This equation is reminiscent of two-scale dilation equations that have been studied in wavelet literature. In particular, these have been analyzed in the past by Daubechies and Lagarias [31] and by Strang [32]. They studied equations of the form shown in equation (5.4).

$$h(x) = \sum_{n=0}^{N} c_n h(\alpha x - \beta_n) \text{ with } \alpha > 1 \tag{5.4}$$

They provided sufficient conditions on the coefficients c_n for a solution $h(x)$ to exist to the above problem. In some sense this is the converse of the problem we are trying to solve, as we are trying to solve for the noise pdf, that is analogous to the coefficients in the two-scale dilation equations. However their results are relevant as we can use them to predict the pdf of the samples of the AR(1) process if we can start with an appropriate noise pdf.

A sufficient condition for the dilation equation to have a solution is described in terms of a function of the sequence of coefficients.

$$P(\omega) = \frac{1}{\alpha} \sum_{n=0}^{N} c_n e^{j\beta_n \omega} \tag{5.5}$$

At most one unique solution, up to normalization, for the two-scale dilation exists if $P(0)=1$ or $\sum_{n=0}^{N} c_n = \alpha$, and in such a case the Fourier transform of this unique solution may be written as in equation (5.6).

$$H(\omega) = H(0)\prod_{j=1}^{\infty} P(\alpha^{-j}\omega) \tag{5.6}$$

The equation for the pdf of the AR(1) process consists of convolution between two continuous functions, however if the noise pdf is discrete (consists of delta functions) then equation (5.3) is identical to the two-scale dilation equation examined in wavelet literature. A discrete noise pdf may be written as in equation (5.7)

$$g_W(x) = \sum_{n=0}^{N} \lambda_n \delta(x-\tau_n) \text{ , with } \sum_{n=0}^{N} \lambda_n = 1 \tag{5.7}$$

In equation (5.7) $\delta(\cdot)$ is the Dirac-Delta function. If we replace this form for the noise in equation (5.3) for the AR process pdf we obtain the following equation.

$$f(x) = \sum_{n=0}^{N} \hat{c}_n f\left(\frac{x}{\rho} - \hat{\tau}_n\right); \ \hat{\tau}_n = \frac{\tau_n}{\rho} \text{ and } \hat{c}_n = \frac{\lambda_n}{|\rho|} \tag{5.8}$$

Equation (5.8) is identical to the two-scale difference equation, when $\rho > 0$. In this case we may use the results from dilation equation theory, and the sufficient condition for the solution $f(x)$ to exist is described in equation (5.9).

$$\sum_{n=0}^{N} \hat{c}_n = \frac{1}{\rho} \text{ or } \sum_{\mathrm{n}=0}^{\mathrm{N}} \frac{\lambda_{\mathrm{n}}}{|\rho|} = \frac{1}{\rho} \tag{5.9}$$

However, as mentioned before, $\sum_{n=0}^{N} \lambda_n = 1$, thereby ensuring that the sufficient condition is always satisfied when $\rho > 0$.

We illustrate the above with some simple examples. For instance if we start with a noise pdf with $\lambda_0 = 0.5, \lambda_1 = 0.5, \tau_0 = -0.25, \tau_1 = 0.25$, we can create an AR(1) process with $\rho = 0.5$ and uniform pdf. We highlight this in Figure 5.13.

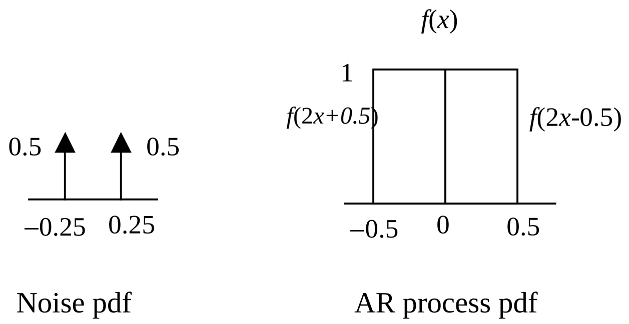

Figure 5.13. Creating AR process with uniform pdf

Similarly we can also create an AR(1) process with $\rho = 0.5$ having the triangle pdf as shown in Figure 5.14.

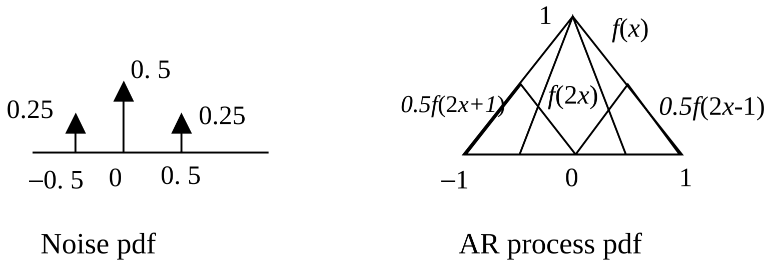

Figure 5.14. Creating AR process with triangle pdf

As shown in Figure 5.14, for the triangle pdf we need to choose our coefficients as $\lambda_0 = 0.25, \lambda_1 = 0.5, \lambda_2 = 0.25, \tau_0 = -0.5, \tau_1 = 0, \tau_2 = 0.5$. In fact, for $\rho = 0.5$, we can create any pdf that may be expressed in terms of multiple convolutions of the uniform pdf with itself, using a discrete noise pdf and we describe this in the following discussion.

All of the above discussion helps us predict the pdf of the AR(1) process if we start with a certain pdf, however we can also examine our original convolution equation and try to solve for the noise pdf that creates a general pdf. We may rewrite equation (5.3) in the frequency domain representation.

$$\psi_f(\omega) = \psi_g(\omega)\psi_f(\rho\omega) \quad \text{or} \quad \psi_g(\omega) = \frac{\psi_f(\omega)}{\psi_f(\rho\omega)} \tag{5.10}$$

In equation (5.10), $\psi_y(\omega) = \Im(y(x))$, the Fourier transform of $y(x)$. These functions $\psi(\cdot)$ are called characteristic functions and are the Fourier transform pairs of the pdfs. Using equation (5.10), we can determine the characteristic function of the noise $\psi_g(\omega)$, given the desired pdf $f(x)$ and hence characteristic function $\psi_f(\omega)$, for the AR(1) process. We can then invert the Fourier transform of the characteristic function $\psi_g(\omega)$ to obtain the pdf of the noise,

$g_W(x)$. We can thus use equation (5.10) to generate some useful pdfs like Gaussian, Exponential and Laplacian as well as the Uniform and Triangle pdf as mentioned before, for AR processes with certain ρ. We show some pdfs that we can create using this process and the corresponding noise pdfs in Table 5-2.

Table 5-2. Noise pdfs needed to create some desired pdfs

Desired pdf	ρ	**Noise pdf (Zero mean)**
Uniform	±0.5	$g_W(x) = 0.5\delta(x+0.25) + 0.5\delta(x-0.25)$
Triangle	±0.5	$g_W(x) = 0.25\delta(x+0.5) + 0.5\delta(x) + 0.25\delta(x-0.5)$
Gaussian	All values	Gaussian
Exponential $f(x) = \begin{cases} e^{-(x+1)}; x > -1 \\ 0 \text{ otherwise} \end{cases}$	$\rho > 0$	$g_W(x) = \begin{cases} \rho\delta(x+1-\rho) + (1-\rho)e^{-(x+1-\rho)}; x \geq -1+\rho \\ 0 \text{ otherwise} \end{cases}$
Laplacian $f(x) = \frac{1}{2}e^{-\lvert x \rvert}$	All values	$g_W(x) = \rho^2\delta(x) + \frac{(1-\rho^2)}{2}e^{-\lvert x \rvert}$

The list of pdfs included in Table 5-2 is not exhaustive and includes only some common pdfs of data. All of the noise pdfs have a mean set to zero and variance $(1-\rho^2)\sigma^2$, where σ^2 is the variance of the AR process. The noise samples can be scaled appropriately to create an AR process with arbitrary variance. We show an illustration of the Noise pdfs for the Exponential and the Laplacian AR processes in the following figure.

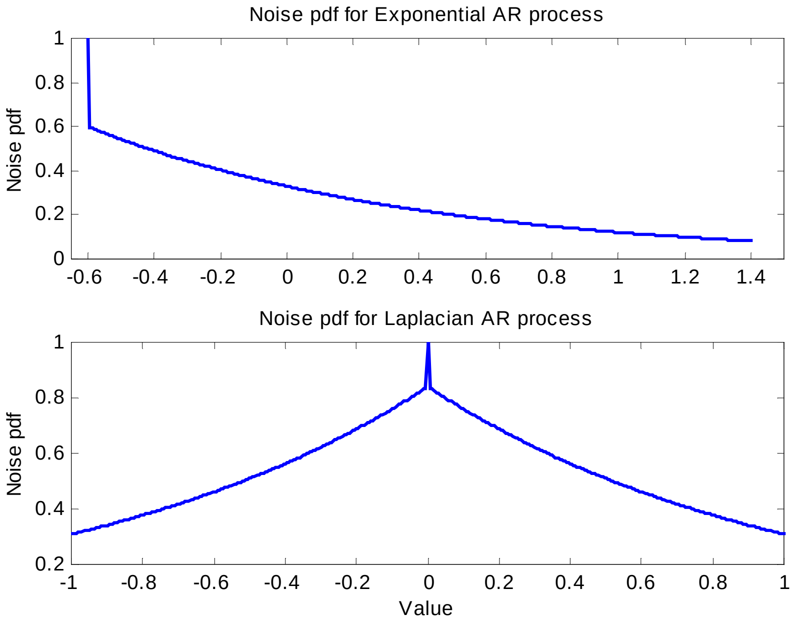

Figure 5.15. Noise pdfs for some AR processes with ρ = 0.4

As is also evident from the table, it is not necessary that all pdfs for an AR process with all different ρ are realizable, as when we invert $\psi_g(\omega)$ it is not necessary that we get a non-negative function, which is a required property of a pdf. For instance, creating the exponential pdf with negative ρ requires the noise pdf to be negative, however this is not possible, and so we cannot create an exponential pdf for an AR process with negative ρ. However it is important to note that if the pdfs $f_1(x)$ and $f_2(x)$ may be created using this procedure, then the pdf $f_3(x) = f_1(x) * f_2(x)$ can always be created using this procedure. This may be shown as follows. Let the characteristic functions for the three processes be $\psi_{fi}(\omega); i = 1,2,3$. Also, let the corresponding noise pdf for the processes be $g_i(x); i = 1,2,3$ with characteristic functions $\psi_{gi}(\omega); i = 1,2,3$. The characteristic function for the noise to create $f_3(x)$ may be written as in equation (5.11).

$$\psi_{g3}(\omega) = \frac{\psi_{f3}(\omega)}{\psi_{f3}(\rho\omega)} = \frac{\psi_{f1}(\omega)}{\psi_{f1}(\rho\omega)}\frac{\psi_{f2}(\omega)}{\psi_{f2}(\rho\omega)} = \psi_{g1}(\omega)\psi_{g2}(\omega) \tag{5.11}$$

When we invert the Fourier transform on both sides of the equation, we have $g_3(x) = g_1(x) * g_2(x)$. Since the convolution of two valid pdfs is also a pdf, this means that we can always create the AR process with pdf $f_3(x)$. We can thus use equation (5.10) and this knowledge to create AR processes with different kinds of pdfs.

5.3. Modeling results with accurate pdf modeling

When we examine the pdfs of the real data traces, we find that they have shapes similar to the exponential pdf with $\rho > 0$ and we show this in Figure 5.17. We may thus use the procedure from Section 5.2 to create appropriate AR processes with Exponential pdfs instead of the Gaussian pdfs. We replace the Gaussian pdf with an Exponential pdf for the Type I Simplified model, described in Section 5.1.2.2, and re-examine the performance of our model. As before, we present results in terms of both the error in autocorrelation function as well as using network simulations.

We first examine the stochastic properties of the trace and measure performance in terms of the error in the autocorrelation function. The autocorrelation functions of the real trace and the trace generated by Type I Simplified model with the Exponential pdf are shown in Figure 5.16.

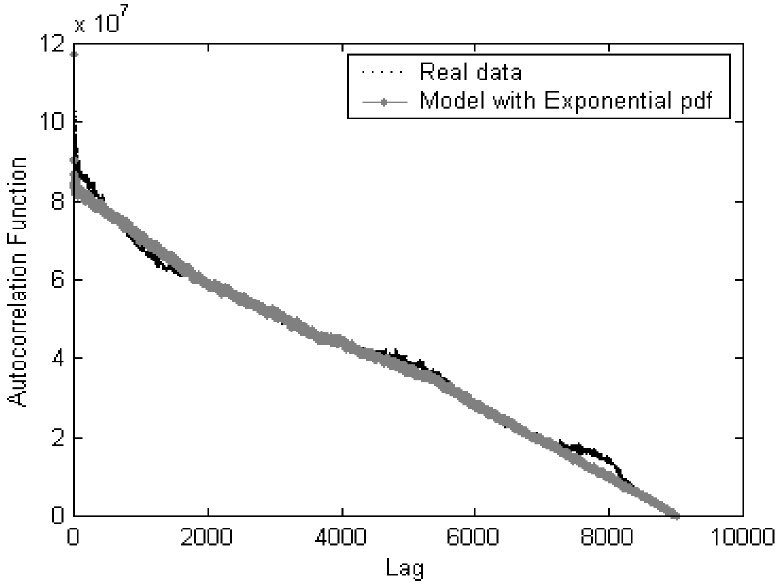

Figure 5.16. Autocorrelation function for real and modeled data for Ads sequence

We can see from Figure 5.16 that the autocorrelation function for the model with the Exponential pdf is quite close to the real autocorrelation function. In terms of root mean squared error in the autocorrelation function, the model with the Exponential pdf has autocorrelation function within 2~4% of the real autocorrelation function. Thus the error is reduced by a factor of 2~3 by actually modeling the pdf of the data. The autocorrelation functions for the individual I, P or B, high, low or medium activity frames are identical for both cases, since the parameters of the AR process are the same in both cases. However when we aggregate these frames by interleaving, the resulting overall autocorrelation function is dependent on the pdf of the data, and not just on the mean and the variance. Thus, modeling the pdf of the data accurately leads to a better modeling of the autocorrelation function. As an illustration we also show the pdf of the high activity B frames from the Ads sequence, the pdf of high activity B frames generated by the model using the Gaussian assumption and using the Exponential pdf in Figure 5.17.

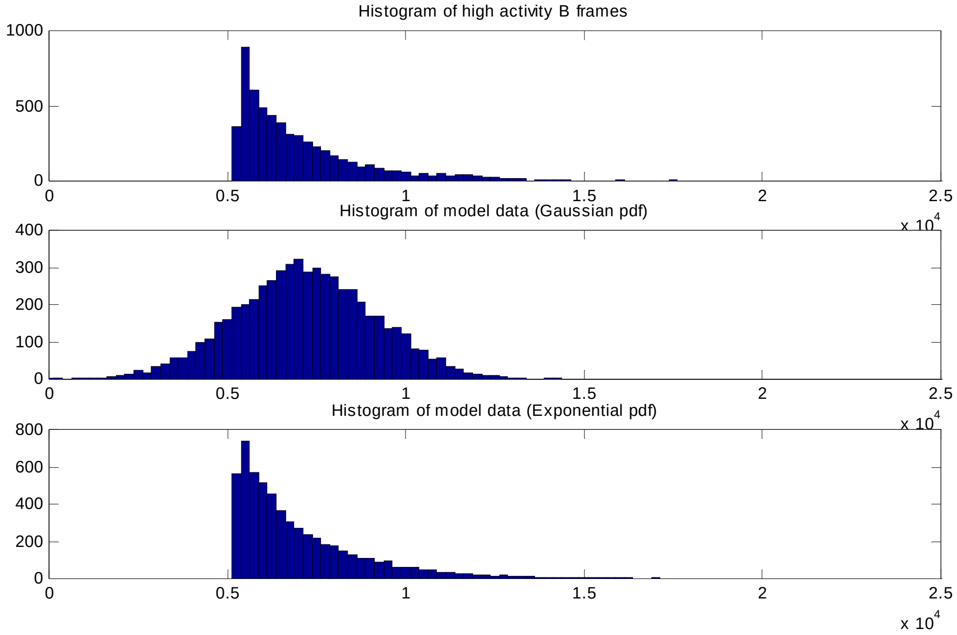

Figure 5.17. pdf of high activity B frame data and data from models

We can see that the model needs to use the exponential pdf to capture the real pdf of the data accurately.

We then use the network simulation to evaluate the performance of the model in predicting the real delay and loss probability. These results are shown for the two sequences in Figure 5.18 and Figure 5.19.

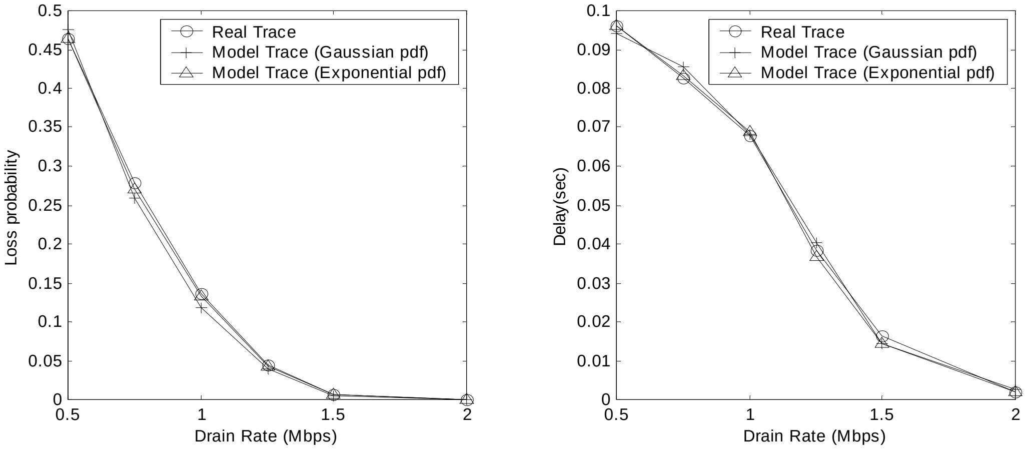

Figure 5.18. Loss probability and delay for Ads sequence

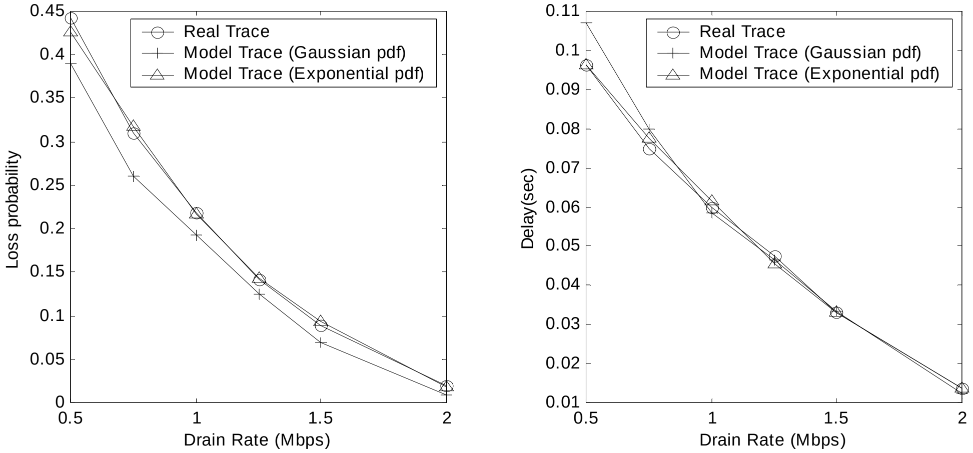

Figure 5.19. Loss probability and delay for News sequence

In the figures, the circles correspond to the delay and loss probability encountered by real data, while triangles correspond to predictions by the model using the Exponential pdf and pluses correspond to predictions by the model using the Gaussian pdf. We can see from the figures that the triangles are closers to the circles than the pluses, which means that actually modeling the pdf of the data improves the performance of the model in predicting delay and loss probability encountered by real data using network simulations. In terms of absolute performance, the loss probabilities and delays predicted by the Exponential model are within 1~3% of the real values as opposed to within 4~15% for the Type I Simplified model, across both the sequences. These predictions are very accurate and these models may be used to estimate network parameters for our conditions under test.

5.4. Conclusion

We propose several models for VBR video sources that allow for a flexible GOP structure, thereby modeling typical traces better. We propose two different kinds of models for data with I, P and B frames. These are the Type I models that choose the type of the frame first before deciding the activity level the frame belongs to and the Type II models that decide the activity level of the frame before choosing the type of the frame. We show that the generated traces are statistically similar to the real data using the error in the autocorrelation function as a measure, which is smaller by a factor of 10~13 over using a fixed GOP model. We also evaluate the performance of the models in terms of predicting the loss probability and delay when we run these traces through a network simulation and show that the delay and loss probabilities predicted by our models are accurate. Our predictions for loss probability and delay are within 4~15% of the loss probability and delay encountered by real traces. The predictions for loss probabilities are

6~18 times smaller in squared error than predictions using a fixed GOP model and our predictions for delay are 3~20 times smaller than the fixed GOP model predictions.

We realize that a Gaussian assumption for the pdf of the data is inaccurate and focus on creating AR processes with the appropriate pdfs to match the data characteristics. We relate the procedure of creating AR processes with appropriate pdfs to solutions of dilation equation from wavelet theory and include solutions for some common pdfs. We then use an Exponential pdf, corresponding to the pdf of the actual data, instead of the Gaussian pdf with our proposed model and re-examine the performance. Performance is measured in terms of both the stochastic properties as well as using simple network simulations. We find that the error in autocorrelation function is an additional 2~3 times smaller for the model using the Exponential pdf as opposed to the model with the Gaussian assumption, which highlights the importance of modeling the pdf of the data accurately. We also see that the prediction for the delay and loss probabilities are between 1~3% of the real values with this more accurate model for the pdf, as opposed to within 4~15% with the Gaussian assumption.

6. Summary and Future Directions

All the work in this thesis is directed toward using stochastic modeling techniques to improve the performance of video coding. We target optimization of video coding in terms of the complexity-quality-bit rate tradeoff and to achieve this goal we focus both on encoder optimizations as well as decoder post-processing. More specifically, we target the mode decisions in the encoding process and error concealment as part of the decoding process. Simultaneously we also realize that these optimizations require information regarding the network condition. In order to get such information from the network we build accurate models for the video traffic so that these may be used to probe the network.

The first contribution of this thesis is in building a classification based framework for making mode decisions to optimize the video encoding. We use features that may be easily computed from the video data to provide an indication of the cost of making a mode decision. The cost may be defined in terms of one or more parameters of the complexity-quality-bit rate tradeoff. We then transform this minimization of cost problem to a more classically understood error probability minimization problem and then use the standard likelihood ratio test to make the optimal decision. This framework is independent of the level at which the mode decision is to be made or the cost that we need to minimize. We illustrate this approach using two common mode decisions in the encoding process, the Intra-Inter decision, made for every 16×16 block in the video frame, and the decision to code or skip a frame in order to meet a certain output target bit rate. We show that using the classification based framework, with the choice of suitable features, we can outperform currently recommended mode decisions in standards.

We use this framework to improve the performance of the Intra-Inter mode decision and reduce the bitstream size by 4.5~4.8% over the mode decision as recommended in the TMN 10 of the H.263 standard. We then use this framework to solve the rate control problem and provide two solutions. The first uses only information from the current frame and the past, and is called the instantaneous mode decision and the second that looks ahead at one future frame before making a decision for the current frame, called the look-ahead mode decision. We show an improvement in performance in the rate-distortion sense over using the current approach both for the instantaneous as well as for the look-ahead mode decision. The improvement in quality, while achieving the same target bit rate for the instantaneous decision is 4~12% while for the look-ahead decision it is 7~18% over the currently proposed rate control. We also extend this work to the scalable video coding and show that with the adaptive SNR/temporal (AST) scalability we improve the performance in terms of quality under error prone conditions by 5~15% over using SNR scalability only.

The second set of contributions of this thesis come from the building of a general stochastic framework for modeling data with large variations. We describe the limitations of the Principal Component Analysis (PCA) in capturing properties of data that consists of multiple clusters and extend the PCA to Mixture of Principal Components (MPC). MPC uses a mixture of eigenspaces to capture the data variations leading to a more efficient representation of such data. We introduce an iterative training algorithm to train these multiple eigenspaces automatically from the data. We also introduce the notion of model based error concealment, where we use a model for a region of interest to replenish any missing data and minimize distortions due to lossy network transmission. Such a model based concealment approach is very useful especially for the MPEG-4 standard, which uses object based coding, thereby making it easy to determine regions of interest and build appropriate models for them.

We use the MPC to capture pose variations in real data and use this model for error concealment under different loss probabilities and target rates of transmission. We show that using such a model based approach to error concealment leads to very good error concealment performance, measured by examining the PSNR as compared to the error free video sequence. We show that model based error concealment provides much better performance by around 5~7 dB over the traditional use of motion compensated residue for concealment. We also show that among the model based concealment schemes, using the MPC as a model leads to better error concealment performance than using the PCA as a model by around 1~2 dB, even with the same number of total parameters.

We realize that this better model for data for variations may be used for tasks other than model based error concealment. We illustrate this fact using two tasks, a face recognition task and a face tracking task. We test recognition over five subjects from the Pose Illumination Expression (PIE) database, exhibiting multiple poses and viewed under different and extreme lighting variations. We show that using the MPC against using the PCA, leads to an improved recognition performance from 83.7% to 95.8%. Thus the MPC may be used to capture pose and lighting variations more efficiently than the PCA, thereby leading to an improved recognition performance. We also illustrate the use of the MPC in foreground background segmentation and the use of the segmentation result to obtain robust tracking performance. It is very important to realize that the MPC may be used to model any kind of data with large amounts of variation thereby making is suitable for any task requiring statistical modeling tools and across any domain, not just the pixel based spatial domain.

The final set of contributions of this thesis lie in the modeling of video traffic. We propose flexible models to capture the different frame types, activity levels and varying scene lengths present in real video traffic data. We use doubly Markov models with autoregressive, AR(1) processes to capture all these data variations. We also attempt to capture the probability density of the data accurately to improve the accuracy of modeling. These models may be used by network designers to estimate the performance of the network under different test conditions, so that they may provide certain performance guarantees. We examine the performance of the models in terms of the stochastic properties of the trace as well as using network simulations. We show that the generated traces are statistically similar to the real data using the error in the autocorrelation function as a measure, which is smaller by a factor of 10~13 over using previously proposed models. We also evaluate the performance of the models in terms of predicting the loss probability and delay when we run these traces through a network simulation and show that the delay and loss probabilities predicted by our models are accurate. Our predictions for loss probability and delay are within 4~15% of the real values. The predictions for loss probabilities are 6~18 times smaller in squared error than predictions using previously proposed models and our predictions for delay are 3~20 times smaller than previously proposed models model predictions.

We realize that a Gaussian assumption for the pdf of the data is inaccurate and focus on creating AR processes with the appropriate pdfs to match the data characteristics. We relate the procedure of creating AR processes with appropriate pdfs to solutions of dilation equation from wavelet theory and include solutions for some common pdfs. We then use an Exponential pdf, corresponding to the pdf of the actual data, instead of the Gaussian pdf with our proposed model and re-examine the performance. We find that the error in autocorrelation function is an additional 2~3 times smaller for the model using the Exponential pdf as opposed to the model with the Gaussian assumption, which highlights the importance of modeling the pdf of the data accurately. We also see that the prediction for the delay and loss probabilities are between 1~3% of the real values with this more accurate model for the pdf, as opposed to within 4~15% with the Gaussian assumption.

All these optimizations are built into the H.263 video codec developed here at CMU to demonstrate the actual coding performance improvements. It needs to be stressed that these optimizations are not specific to the H.263 standard and may easily be adapted to a video codec employing a different coding standard. The H.263 standard was used as it provided a ready platform to evaluate these algorithms.

There are some interesting extensions for the work described in this thesis. For all of our work in this thesis, we do not actually transmit the data over real networks, but use network simulations to gather the results. It is of interest to be able to use these optimizations over a real network scenario to evaluate the performance. Some other issues that are interesting in such a scenario include greater cooperation between the encoder and the decoder, via a feedback channel in order to further improve the performance of the system. The MPC that we developed in this thesis may be used for any other applications requiring modeling data. For instance they may be used to capture expression variations in human faces. The mixtures may also be extended to account for the likelihood of the data belonging to multiple clusters.

Bibliography

[1] CMU H.263 Video Codec. For more information please see http://amp.ece.cmu.edu.

[2] C. –T. Chen and A. Wong, “A self-governing rate buffer control strategy for pseudoconstant bit rate video coding,” *IEEE Trans. Image Processing*, vol. 2, pp. 50-59, January 1993.

[3] J. Choi and D. Park, “ A stable feedback control of the buffer state using the controlled multiplier method,” *IEEE Trans. Image Processing*, vol. 3, no. 5, pp. 546-58, September 1994.

[4] C. Y. Hsu, A. Ortega and A. R. Reibman, “Joint selection of source and channel rate for VBR video transmission under ATM policing constraints,” *IEEE J. Selected Areas in Commun.*, vol. 15, no. 6, August 1996.

[5] H. Song, J. Kim and C. –C. Jay Kuo, “Real-time encoding frame rate control for H.263+ video over the internet,” *Signal Processing: Image Communication*, no. 15, September 1999.

[6] F. C. Martins, W. Ding and E. Feig, “Joint control of spatial quantization and temporal sampling for very low bit rate video,” IEEE International Conference on Acoustics, Speech, and Signal Processing, 1996.

[7] T. Chiang and Y. –Q. Zhang, “A new rate control scheme using quadratic rate distortion model,” *IEEE Trans. Circuits Syst. Video Technol*., vol. 7, no.1, p. 246-50, September 1997.

[8] W. Ding and B. Liu, “Rate control of MPEG video coding and recording by rate-quantization modeling,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 12-20, February 1996.

[9] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. Wiley and Sons, New York, NY, 1973.

[10] T. Hastie and W. Stuetzle, “Principal curves,” *Journal of the American Statistical Association*, vol. 84, pp. 502-16, 1989.

[11] E. Oja, “Neural networks, principal components and subspaces,” *International Journal of Neural Systems*, vol. 1, pp. 61-68, 1989.

[12] S. Y. Kung and K. I. Diamantaras, “A neural network learning algorithm for adaptive principal component extraction (APEX),” Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 861-64, 1990.

[13] T. Cox and M. Cox, *Multidimensional Scaling*, Chapman and Hall, London 1994.

[14] S. T. Roweis and L. K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *Science*, vol. 290, pp. 2323-26, December 2000.

[15] N. Kambhatla and T. K. Leen, “Dimension reduction by local principal component analysis,” *Neural Computation*, vol. 9, pp. 1493-1516, 1997.

[16] M. Tipping and C. M. Bishop, “Mixtures of probabilistic principal component analyzers,” *Neural Computation*, vol. 11, no. 2, pp. 443-82, 1999.

[17] S. Aign and K. Fazel, “Temporal and spatial error concealment techniques for hierarchical MPEG-2 video codec,” Proc. Globecom, p. 1778-83, 1995.

[18] H. Sun and W. Kwok, “Concealment of damaged block transform coded images using projections onto convex sets,” *IEEE Trans. Image Proc.*, vol. 4, no. 4, p. 470-77, April 1995.

[19] M-J. Chen, L-G. Chen and R-M. Weng, “Error concealment of lost motion vectors with overlapped motion compensation,” *IEEE Trans. Circuits Syst. Video Technol*., vol.7, no. 3, p. 560-63, June 1997.

[20] L. Atzori, F. G. B. De Natale, “Error concealment in video transmission over packet networks by a sketch based approach,” *Signal Processing: Image Communication*, vol. 15, pp. 57-76, 1999.

[21] T. V. Lakshman, A. Ortega and A. R. Reibman, “VBR: Video Tradeoffs and Potentials,” *Proceedings of the IEEE*, vol. 86, no. 5, pp. 952-73, 1998.

[22] B. Maglaris, D. Anastassiou, P. Sen, G. Karlsson and J.D. Robbins, “Performance models of Statistical Multiplexing in Packet Video Communication,” *IEEE Trans. Comm.*, vol. 36, pp. 834-43, 1988.

[23] P. Sen, B. Maglaris, N, Rikli and D. Anastassiou, “Models for Packet Switching of Variable-Bit-Rate Video Sources,” *IEEE J. on Select. Areas in Comm.*, vol. 7, no. 5, June 1989.

[24] F. Yegenoglu, B. Jabbari and Ya-Qin Zhang, “Motion-classified Autoregressive Modeling of Variable Bit Rate Video,” *IEEE Trans. CSVT*. vol. 3, no. 1, Feb 1993.

[25] M. Izquierdo and D. Reeves, “A survey of statistical source models for variable-bit-rate compressed video,” Multimedia Systems, vol.7, no.3, pp. 199-213.

[26] N. Doulamis, A. Doulamis and S. Kollias, “Modeling and Adaptive Prediction of VBR MPEG Video Sources,” pp. 27-32, 1999 IEEE Third Workshop on Multimedia Signal Processing, September 1999, Copenhagen, Denmark.

[27] K. Chandra and A. Reibman, “Modeling One- And Two-Layer Variable Bit Rate Video,” *IEEE/ACM Trans. Networking*, vol. 7, no. 3, pp. 398-413.

[28] D. Turaga and T. Chen, “Modeling of Dynamic Video Traffic,” International Symposium on Circuits and Systems, May 2000.

[29] D. Turaga and T. Chen, “Activity-Adaptive Modeling of Dynamic Multimedia Traffic,” International Conference on Multimedia and Exposition, August 2000.

[30] D. Turaga and T. Chen, “Hierarchical modeling of variable bit rate video sources,” Packet Video Workshop, May 2001.

[31] I. Daubechies and J. Lagarias, “ Two-scale difference equations. 1. Existence and global regularity of solutions,” *SIAM J. Math Anal.*, vol. 22, no. 5, pp. 1388-1410, September, 1991

[32] G. Strang, “Wavelets and Dilation Equations: A Brief Introduction,” *SIAM Review*, vol. 31, no. 4, pp. 614-627, December 1989.

[33] Ahmed, N., Natarajan, T., and Rao, K. R., “Discrete cosine transform,” *IEEE Trans. on Computers*, C-23: 90-3, 1974.

[34] Rao, K. R., and Yip, P., *Discrete Cosine Transform*, Academia Press, New York, 1990.

[35] Netravali, A. N., and Robbins, J. D.,” Motion-compensated television coding Part I,” *Bell Systems Technical Journals*, v. 58(3), pp. 631-670, March 1979.

[36] Netravali, A. N., and Haskell, B. G., *Digital Pictures*, Plenum Press, New York and London, 1995, 2nd ed.

[37] D. S. Turaga and T. Chen, “Estimation and Mode Decision for Spatially Correlated Motion Sequences,” to be published in *IEEE Trans. Circuits Syst. for Video Technol.*, 2001.

[38] D. Turaga and T. Chen, Book Chapter “Fundamentals of Video Compression: H.263 as an Example”, *Compressed Video Over Networks*, The Signal Processing Series, Marcel Dekker, 2000.

[39] ITU-T Recommendation H.261: “Video codec for audiovisual services at p x 64 kbit/s,” Geneva, 1990, revised at Helsinki, March 1993.

[40] *Video Coding for Low Bit rate Communication*, ITU-T Recommendation H.263 Version 2, Jan. 1998.

[41] Motion Pictures Experts Group, "Overview of the MPEG-4 Standard", ISO/IEC JTC1/SC29/WG11 N2459, 1998.

[42] Video Encoder Test Model, Near-Term, Version 10 (TMN10) Draft 1, Apr. 1998.

[43] MPEG-4 Video Group, “MPEG-4 Verification Model v14.0,” ISO/IEC N2932, Oct. 1999.

[44] G. McLachlan and T. Krishnan, “The EM algorithm and extensions,” Wiley Interscience, New York, NY, 1996.

[45] S. Wolf and M. H. Pinson, “Spatial-Temporal Distortion Metrics for In-Service Quality Monitoring of Any Digital Video System,” SPIE International Symposium on Voice, Video, and Data Communications, Boston, MA, September 11-22, 1999.

[46] G. D. Forney, Jr., “The Viterbi Algorithm,” *Proc. IEEE*, vol. 61, pp. 268-78, March 1973.

[47] D. S. Turaga and T. Chen, “Classification Based Mode Decisions for Video Over Networks,” under revision for publication in *IEEE Trans. Multimedia*, Special Issue on Multimedia over IP.

[48] M. I. Sezan and M. Tekalp, “Adaptive image restoration with artifact suppression using the theory of convex projections,” *IEEE Trans. Acoust. Speech and Signal Processing*, vol. 38, no. 1, pp. 181-5, January 1990.

[49] J. Huang and T. Chen, "Tracking of Multiple Faces for Human-Computer Interfaces and Virtual Environments", IEEE Intl. Conf. on Multimedia and Exposition, New York, August 2000.

[50] M. Yajnik, S. Moon, J. Kurose, D. Towsley, “Measurement and modeling of the temporal dependence in packet loss,” Proc. IEEE INFOCOM, pp. 345-52, March 1999.

[51] M. Turk and A. Pentland, “Eigenfaces for recognition,” *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71-86, 1991.

[52] X. Liu, T. Chen and B.V.K. Vijaya Kumar, “Robust face authentication for multiple subjects using eigenflow,” Technical Report AMP 01-05, Advances Multimedia Processing Lab, Carnegie Mellon University, 2001.

[53] T. Sim, S. Baker and M. Bsat, “The CMU pose, illumination and expression (PIE) database of human faces,” Technical Report CMU-RI-TR-01-02, Robotics Institute, Carnegie Mellon University, January 2001.