Position-Dependent Encoding

by

Aleksandar Pfajfer

B.S., Electrical Engineering
Massachusetts Institute of Technology, 1992

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the
Degree of

Master of Science
in Electrical Engineering

at the
Massachusetts Institute of Technology

May 1994

Signature of Author ______________________________
Department of Electrical Engineering
May 6, 1994

Certified by ______________________________
Jae S. Lim
Professor of Electrical Engineering
Thesis Supervisor

Accepted by ______________________________
Frederic R. Morgenthaler
Chairman, Committee on Graduate Study

Position-Dependent Encoding

by

Aleksandar Pfajfer

Submitted to the Department of Electrical Engineering and Computer Science
on May 6, 1994 in Partial Fulfillment of the Requirements for
the Degree of Master of Science in Electrical Engineering

Abstract

In typical video compression algorithms, the DCT is applied to the video, and the resulting DCT coefficients are quantized and encoded for transmission or storage. Most of the DCT coefficients are quantized to zero. Efficient encoding of the DCT coefficients is usually achieved by encoding the location and the amplitude of the non-zero coefficients. Since in typical MC-DCT compression algorithms, up to 90% of the available bit rate is used to encode the locations and the amplitudes of the non-zero DCT coefficients, efficient encoding of the location and amplitude information is extremely important for high quality compression.

A novel approach to encoding of the location and amplitude information, the position-dependent encoding, is being examined. Position-dependent runlength encoding and position-dependent amplitude encoding exploit the inherent differences in statistical properties of the runlengths and amplitudes as a function of position within the 8x8 DCT coefficient block. This novel method is being compared to the classical single-codebook separate runlength and amplitude encoding, as well as to the single-codebook joint runlength / amplitude encoding.

Thesis Supervisor: Jae S. Lim

Title: Professor of Electrical Engineering

To mama and tata
for their love and support.

Acknowledgments

I would like to express my gratitude to Professor Jae Lim for his guidance, advice and motivation throughout my thesis work. Many thanks for friendship and support to everybody at the Advanced Television and Signal Processing Group at MIT, all the graduate students and especially Cindy LeBlanc. Special thanks to Lon Sunshine for reviewing the original manuscript. My deepest gratitude goes, however, to John Apostolopoulos, for starting the research on ideas behind this thesis and teaching me what he learned, for his enthusiastic and thorough discussions on video compression, and many helpful suggestions for my thesis work and manuscript preparation.

The research reported in this thesis was sponsored principally by the Advanced Telecommunications Research Program (ATRP) at MIT. Consortium members that supported the research are Ampex, Capital Cities/ABC, Eastman Kodak, General Instrument, IBM, Motorola, Polaroid, and PBS.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The objective of digital compression systems is to reduce the amount of information available to them in order to transmit it or store it within a constrained medium, while keeping the fidelity at an acceptable level. Video compression systems used for transmission purposes, such as High Definition Television systems, have a very constrained medium: bandwidth. Therefore, it is crucial to achieve the highest quality possible within as few bits as possible. In typical MC-DCT compression schemes, the quantized DCT coefficients take up to 90% of the bits. Efficient encoding of the DCT coefficients is then of utmost importance to the quality of the compression, since any improvement in the coding performance will yield significant improvements to the overall system.

Before discussing the encoding of the quantized DCT coefficients in Chapter 2, it is appropriate to begin, in this chapter, with a brief overview of video compression systems.

Video Compression Systems

A typical video compression system consists of three distinct but interrelated stages: representation stage, quantization stage, and encoding stage (Figure 1-1).

The representation stage performs the necessary transformations on the incoming video signal to minimize the correlation that exists along the three dimensions of the signal (color, temporal, and spatial). The most commonly used transformations are: the color space conversion from RGB (red, green, blue) to YUV (luminance and two chrominance components); motion-compensated prediction along the temporal dimension; and the 8x8 spatial-block Discrete Cosine Transform (DCT) from the spatial domain of the video into the frequency domain.

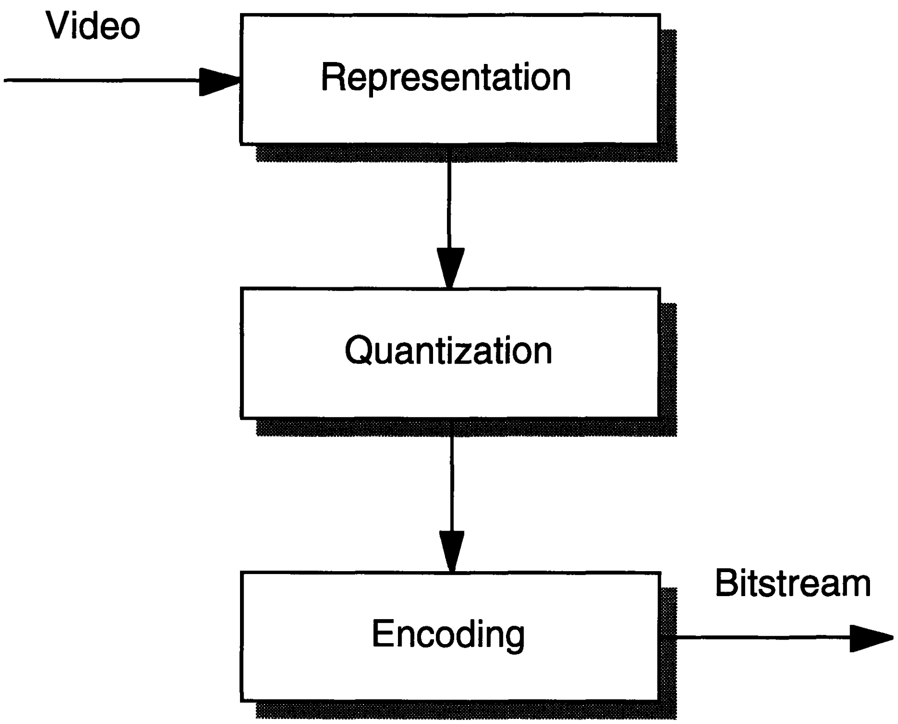


Figure 1-1: A typical video compression system.

The quantization stage performs the actual selection and reduction of the information content of the video, in order to enable digital transmission or storage within the available bandwidth or memory. Usually incorporated within the quantization stage is a model of the human visual system. For example, the human eye is more sensitive to low-frequency quantization noise than to high-frequency quantization noise; therefore, the low-frequency coefficients are quantized more finely then the high-frequency coefficients.

The encoding stage converts the quantized coefficients into an outgoing bit stream to be transmitted or stored. A variety of entropy coding schemes may be used, such as Huffman coding or arithmetic coding, to minimize the average number of bits required for coding of the quantized coefficients. A number of factors have led to Huffman coding being the near-universal choice for image and video compression algorithms today.

For a description of general video compression approaches please refer to [1,2]. For an in-depth look at the techniques mentioned above see [3,4].

Constant Bit Rate Constraint

Video compression systems typically operate at either a constant quantization level, such as assumed in Figure 1-1, or at a constant outgoing bit rate, in which case an implementation as shown in Figure 1-2 is required. Compression systems with constant quantization levels have a variable bit rate, while systems with a constant bit rate constraint must incorporate adaptive quantization in order to keep the bit rate constant. Many compression applications, and in particular most of the broadcast scenarios, have the constant bit rate constraint. In these situations it is particularly important to perform the adaptive quantization in such a manner as to maintain the highest video quality while satisfying the constant bit rate constraint.

Adaptive quantization may be performed on either a frame by frame basis (global buffer control), or on a block by block basis (local buffer control). The buffer keeps track of the aggregate bit rate for the last few frames or blocks and tries to keep it within some tolerable (predetermined) bounds around the target bit rate. If there is buffer overflow at the current quantization level, the current frame or block is quantized more and more coarsely until the aggregate bit rate falls within the predetermined bounds. Conversely, if there is buffer underflow, the current frame or block is quantized more and more finely until the aggregate bit rate falls within the predetermined bounds. Note that with a finite number of quantization levels, the constant bit rate can be achieved only approximately. Therefore, the bounds around the target bit rate are necessary as a decision criterion for stopping the adaptive quantization process on the current frame or block.

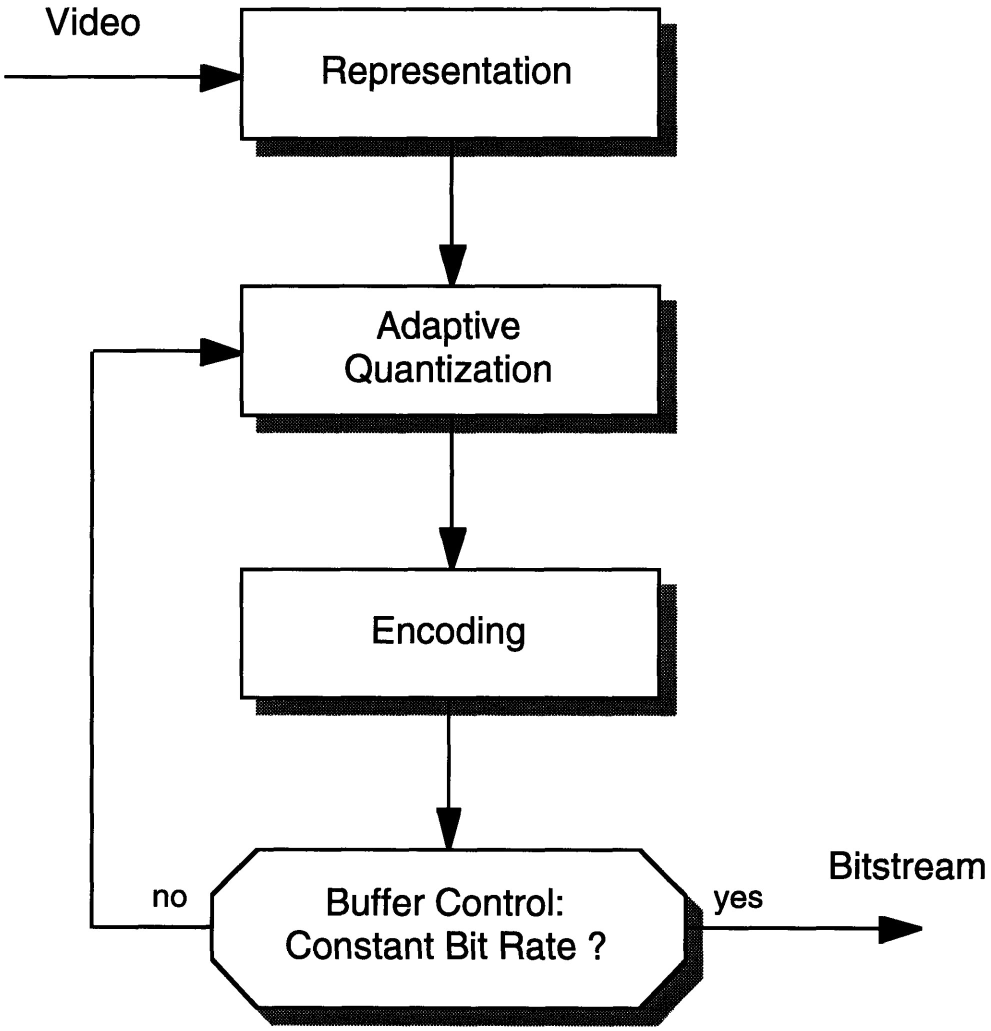


Figure 1-2: A video compression system with a buffer and a feedback loop added, in order to maintain a constant bit rate.

A possible implementation of a constant bit rate video compression system, such as High Definition Television, is presented in Figure 1-3, integrating the techniques discussed so far in this chapter. The general scheme presented is the one used in both the MIT/GI Channel-Compatible DigiCipher digital HDTV system [4], and the MPEG2 video coding standard [5,6].

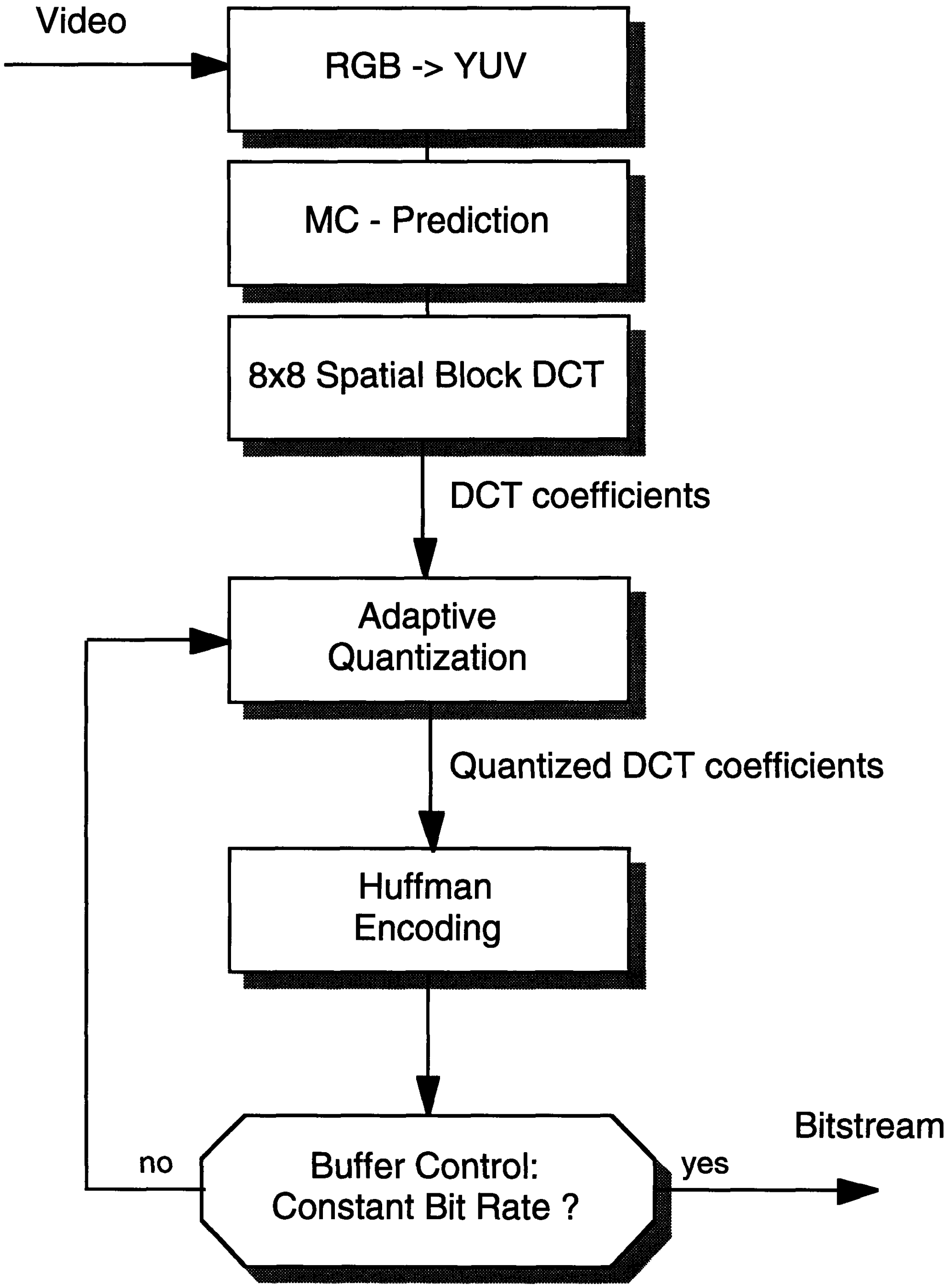


Figure 1-3: A constant bit rate video compression system, such as for HDTV.

In typical applications of the constant-bit-rate video compression systems almost 90% of the available bit rate is used for coding the DCT coefficients (the rest being used for motion vectors and overhead information). Therefore, a large gain in coding efficiency can be obtained through better coding of the DCT coefficients. For that reason, efficient coding of the DCT coefficients is highly important.

The next chapter (Chapter 2) explains the conventional approaches to coding of the quantized DCT coefficients. A new scheme for encoding the DCT coefficients, the position-dependent encoding (PDE), is presented in Chapter 3. Chapter 4 introduces escape codes into the PDE. The results of comparisons to other encoding schemes are summarized in Chapter 5 to illustrate the superior performance of the PDE scheme. Finally, Chapter 6 presents conclusions as well as suggestions for further research.

The research presented was conducted specifically with HDTV in mind. Therefore, this thesis concentrates on the compression of video, and incorporates methods for coding both intra- and inter-mode DCT blocks. Considering that the coding of still images involves only intra-mode DCT blocks, the results obtained in video coding are immediately extendible to image coding.

Although the results to be presented are based on a compression scheme using the Block DCT, the ideas developed are also applicable to any other compression scheme based on a spatial-domain to frequency-domain transformation (e.g. wavelet transform, subband transform). Furthermore, the general concepts are also applicable to other compression applications, such as speech or audio.

Chapter 2

Conventional Coding Approaches

The encoder in Figure 1-3 receives a sequence of 8x8 blocks of quantized DCT coefficients, such as the one in Figure 2-1. In typical video compression applications, and specifically in HDTV, most of the DCT coefficients in a block are quantized to zero, therefore, producing a sparse 8x8 matrix. An efficient way to encode the information contained in a sparse matrix is to encode the location and amplitude of only the non-zero coefficients.

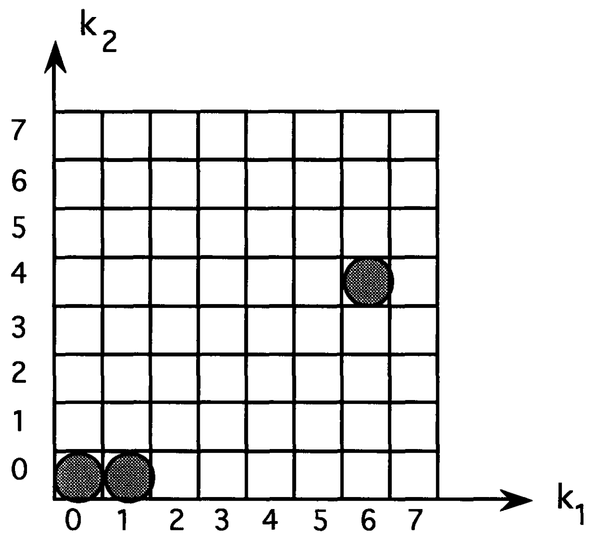


Figure 2-1: An example 8x8 block of quantized DCT coefficients. Non-zero coefficients are shaded. k_1 is the horizontal frequency, k_2 the vertical frequency; DC is in the bottom left corner.

Zigzag Scanning and Runlength Coding

One of the most frequently used approaches to encoding the location and amplitude information is illustrated in Figure 2-2. The quantized coefficients are ordered into a one-dimensional vector, such as through zigzag scanning of the block, starting at coefficient (0,0) (i.e. DC), and finishing at coefficient (7,7). The locations of the non-zero coefficients are described by encoding the runlengths, that is the runs of zero coefficients between the subsequent non-zero coefficients in this particular ordering of coefficients. The first coefficient after a non-zero coefficient is considered the starting position of the appropriate runlength. In addition, the amplitudes of non-zero coefficients are encoded.

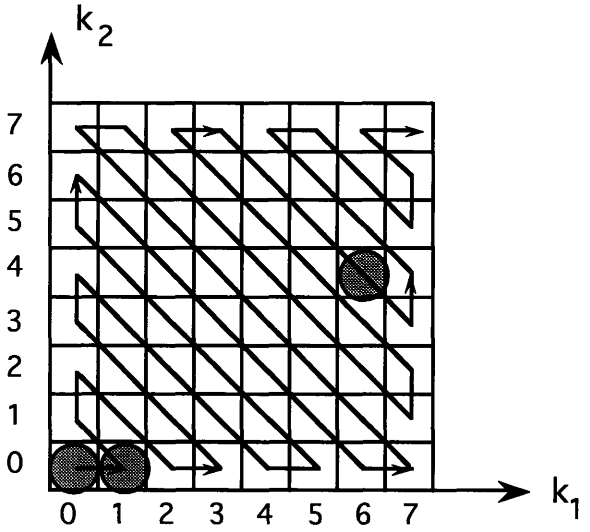


Figure 2-2: Zigzag scanning of the DCT block and ordering of the quantized coefficients. The scan starts at DC (0,0) and ends at (7,7).

For example, the sequence of events to be encoded in Figure 2-2 is:

(1) runlength 0;

(2) amplitude of (0,0) coefficient;

(3) runlength 0;

(4) amplitude of (1,0) coefficient;

(5) runlength 50;

(6) amplitude of (6,4) coefficient; and

(7) EOB (End of Block) "runlength".

The EOB event signifies that there are no more non-zero coefficients in this block. Although EOB is technically not a runlength, it is treated as such for coding purposes. The above scheme is, actually, used for the inter-mode blocks only. In the intra-mode blocks the amplitude of the DC coefficient is always coded since it is almost never zero. So, if the block in Figure 2-2 were an intra-mode one, the sequence of events to be encoded would be the following:

(1) amplitude of (0,0) coefficient;

(2) runlength 0;

(3) amplitude of (1,0) coefficient;

(4) runlength 50;

(5) amplitude of (6,4) coefficient; and

(6) EOB.

Huffman Coding

In order to minimize the average bit rate, encoders in typical compression systems use Huffman codebooks. Huffman codes are a type of entropy codes [1], where variable length codewords are used. Huffman codes exploit the statistics of the data to be encoded by assigning shorter codewords to the events that are more likely, and longer codewords to the events that are less likely. The probability distributions that are necessary for creating the Huffman codebooks are usually obtained by collecting the relevant statistics from a set of video sequences (the training set).

Separate and Joint Huffman Codebooks

The runlengths and amplitudes can be treated as separate events, in which case one codebook is used to encode the runlengths, and another codebook is used to encode the amplitudes. This approach is henceforth referred to as the **separate coding** of runlengths and amplitudes. Obviously, both codebooks are one-dimensional.

On the other hand, a runlength and the following amplitude can be treated jointly as a single event, in which case only one codebook is needed. However, this codebook is two-dimensional. The advantage of the **joint coding** approach (which is the MPEG2 standard [5]) is that it exploits the correlation between a runlength and the following amplitude. As will be illustrated in Chapter 3, short runlengths are usually followed by large amplitudes, and, conversely, long runlengths are usually followed by small amplitudes.

Differences in Statistics and Motivation for PDE

In typical video low-frequency components usually have higher energy than the high-frequency components. Also, because of the properties of the human visual system that are incorporated within the quantization stage (see discussion in Chapter 1), the compression scheme places more importance on the low-frequency DCT coefficients than on the high-frequency DCT coefficients. As a consequence of these two factors, the non-zero quantized coefficients are usually concentrated in the low-frequency region of the 8x8 blocks, and are very sparse throughout the high-frequency region. Therefore, it is very likely, that in the zigzag ordering of the DCT coefficients, a low-frequency non-zero coefficient is followed by another non-zero coefficient, producing a 0 runlength, as in the example of Figure 2-2. Conversely, the most likely event after a high-frequency non-zero coefficient is EOB (i.e. no more non-zero coefficients).

Because of the finer quantization of the low-frequency DCT coefficients, as well as because of the fact that they have higher energy to begin with, they may have large amplitudes. In the high-frequency region, the coefficients have generally smaller energy and are quantized more coarsely, and, therefore, are most likely small.

The position-dependent encoding scheme, that is to be discussed in Chapter 3, exploits the differences in statistics of runlengths and amplitudes as a function of frequency (i.e. the position in the 8x8 block), as mentioned above. The two schemes described previously in this chapter (single-codebook separate and single-codebook joint runlength and amplitude encoding) do not exploit these inherent differences in statistics, since the codebooks they use are designed based on the average statistics of all the non-zero quantized DCT coefficients in the block. In essence, the single-codebook schemes are designed to perform reasonably well everywhere in the block.

Although the position-dependent encoding approach to be discussed is developed as an extension of separate encoding of runlengths and amplitudes, the idea behind it is equally applicable to developing a position-dependent extension of joint encoding of runlengths and amplitudes.

Chapter 3

Position-Dependent Encoding

The position-dependent encoding (PDE) is developed on the basis of the separate coding of runlengths and amplitudes. The PDE introduces multiple codebooks based on the starting position of the runlength, or the position of the non-zero coefficient, in order to exploit the differences in range and statistics of runlengths and amplitudes as a function of their position in the block [7,8]. The runlength encoding and the amplitude encoding are discussed in Section 3.1 and Section 3.2, respectively. Section 3.3 discusses the differences in statistics between the intra-mode blocks and the inter-mode blocks, as well as between the luminance component and the chrominance components.

3.1 Position-Dependent Runlength Encoding

Position-dependent runlength encoding exploits the differences in range and statistics of the runlengths as a function of the starting position of the runlength within the 8x8 block. The differences are illustrated using Figure 3-1.

Because the compression system places more importance on low-frequency DCT coefficients, the non-zero quantized coefficients are concentrated in the low-frequency region. Therefore, it is very likely that a non-zero low-frequency coefficient (such as DC) will be followed by another non-zero coefficient (i.e. coefficient (1,0)), resulting in 0 runlength. If this were an intra block, the above runlength would be the first runlength event. If it were an inter block, it would be the second runlength event, with the first runlength event being a 0 runlength as well (since the first non-zero coefficient is at DC). A high-frequency coefficient, such as (6,4), is most likely the last non-zero coefficient of the block, therefore it most likely followed by an EOB "runlength". As was explained in Chapter 2, although EOB is not a runlength, it is treated as such for coding purposes, and is, therefore, an entry in the runlength codebooks.

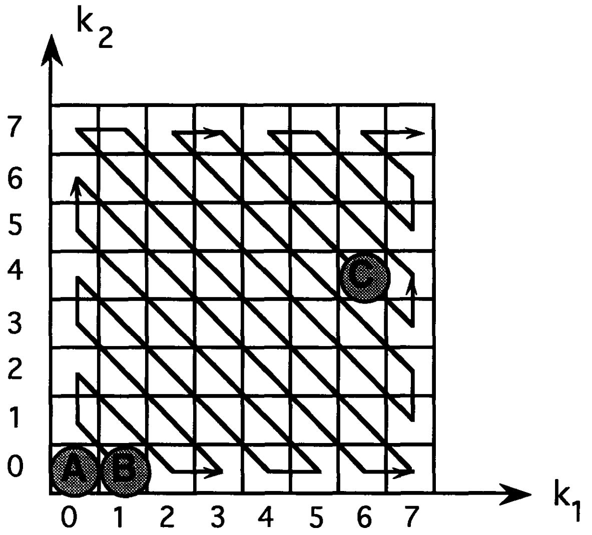


Figure 3-1: Statistics of the runlengths depend on their starting position within the block.

The range of runlengths possible at a certain starting position depends on where that starting position is within the zigzag order. For example, the runlengths starting at position (1,0) (i.e. after the non-zero DC coefficient) can be anywhere between 0 and 62. The runlengths starting at position (7,3) (i.e. after the non-zero coefficient at (6,4)) can be between 0 and 10. EOB is of course a possibility after either of the two non-zero coefficients mentioned above. So, even if a uniform encoder (i.e. an encoder with uniform-length codeword assignment) were used, the number of bits necessary to encode all the runlength possibilities (i.e. the length of the uniform-length codewords) would decrease as a function of the position within the block. For the runlengths starting at first

32 positions in the zigzag ordering, 6 bits are required to encode all the possibilities distinctly, while for the next 16 starting positions only 5 are needed, and so on.

These observations lead to the idea of using one codebook for coding the runlengths starting at position (1,0) and another codebook for coding the runlengths starting at (7,3). In fact, the whole 8x8 block can be partitioned into regions which use different codebooks. The coefficients are grouped based on the similarities in statistics of the runlengths starting at those positions. An example assignment of codebooks is shown in Figure 3-2. The runlengths starting at the positions marked with identical patterns share codebooks.

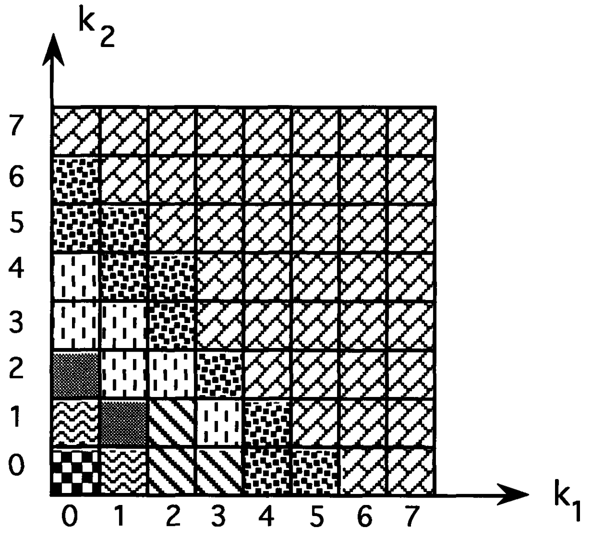


Figure 3-2: An example assignment of runlength codebooks.

3.2 Position-Dependent Amplitude Encoding

Position-dependent amplitude encoding exploits the differences in statistics of amplitudes of the non-zero quantized DCT coefficients as a function of their position within the 8x8 block. The differences are illustrated using Figure 3-3.

The video compression system places more importance on the low-frequency coefficients than on the high-frequency coefficients, i.e. the low-frequency coefficients are more finely quantized than the high-frequency ones. Also, the low-frequency coefficients typically have higher energy than the high-frequency coefficients to begin with. Therefore, the vast majority of the non-zero quantized high-frequency coefficients has small amplitudes, as opposed to the low-frequency coefficients, which may have large as well as small amplitudes. For example, the coefficient (0,0) may be large as well as small, while the coefficient (6,4) is most likely going to be small.

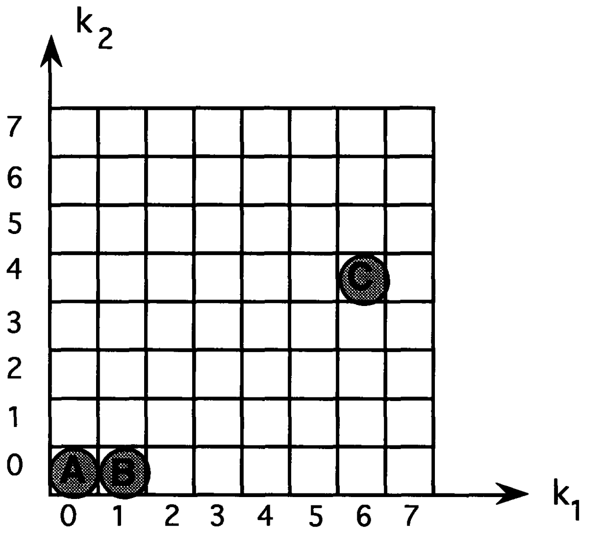


Figure 3-3: Statistics of the amplitudes depend on their position within the block.

This observation leads to the idea of using one codebook for coding the amplitudes at (0,0), and another one for coding the amplitudes at (6,4). In fact, the whole 8x8 block can be partitioned into regions which use different codebooks. The coefficients are grouped based on the similarities in statistics of amplitudes of the non-zero quantized DCT coefficients at those positions. An example assignment of codebooks is shown in Figure 3-4. The non-zero quantized DCT coefficients at positions marked with identical patterns share codebooks.

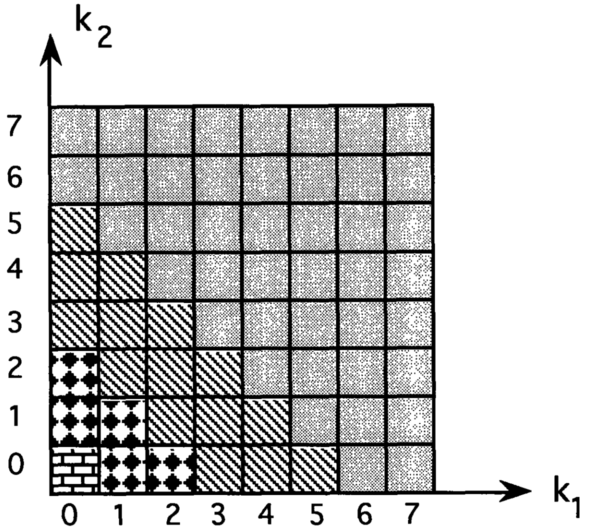


Figure 3-4: An example assignment of amplitude codebooks.

3.3 Intra vs. Inter and Luminance vs. Chrominance

The runlength and amplitude statistics do not only depend on their position in the 8x8 quantized DCT coefficient block, but also on what type of block they come from. The statistics are different depending on whether the block is intra-mode or inter-mode, and whether it is in the luminance component (Y) or in one of the chrominance components (U or V).

The non-zero quantized DCT coefficients in the intra blocks are highly concentrated in the low-frequency regions of the blocks. The inter blocks, on the other hand, correspond to the MC-prediction residual, so they tend to have significant mid- and high-frequency content, as well as low-frequency content. Consequently, the coefficients are spread more evenly and sparsely throughout the block. Obviously, this results in significantly different runlength and amplitude statistics in intra and inter blocks.

Significant differences also exist between the luminance and chrominance components. Since the human visual system has a reduced frequency response to the chrominance components as opposed to the luminance component, typical compression systems low pass filter and subsample the chrominance by a factor of 2 along both spatial directions (4:1:1 subsampling). This operation results in different statistics for the runlengths and amplitudes for the chrominance as compared to the luminance. The two chrominance components have similar statistics.

These observations motivate introduction of different sets of codebooks for each of the following four types of blocks for both runlengths and amplitudes: intra Y, intra UV, inter Y, and inter UV.

It is interesting to note that the 4:1:1 subsampling results in four luminance blocks for every two chrominance blocks (one block for each of the two chrominance components). Most of the bit rate is then occupied by the luminance bits. This fact suggests that exploiting differences in statistics in luminance blocks is more important then exploiting the analogous differences in the chrominance blocks, since the potential overall gain is higher for the luminance. As will be discussed in Chapter 5 and illustrated in Appendix A, more codebooks are used for coding of the luminance component than for coding of the chrominance components exactly because of the higher importance of the luminance component in the bit rate.

Chapter 4

Practical Considerations

An escape code is a sequence of bits introduced into the bitstream to signify that the following event is not coded using a codeword from a Huffman codebook, but using a corresponding uniform-length codeword. There are two practical issues that motivate the introduction of escape codes. The first issue is the fact that both the complexity of the implementation and the size of the memory required to store the codebooks increase dramatically for the PDE approach as compared to the two single-codebook approaches. The increase in the complexity and the memory requirement are due to the increased number of codebooks used in the PDE.

The second issue is the inability to come up with the perfect set of runlength and amplitude statistics. Namely, the statistics used to design the Huffman codebooks are collected from a training set of sequences. An event that did not occur in the training set will end up with an exceedingly long Huffman codeword. As long as the test sequences conform to the statistical model used to design the Huffman codebooks, the scheme performs well. However, if an event that is deemed unlikely by the Huffman codebook actually occurs, the scheme performs poorly (i.e. the local bit rate increases dramatically). This of course can happen with the single-codebook encoding as well. However, the probability is higher for the PDE scheme since the same amount of training data is divided up between different codebooks. In a smaller data set there are more events that do not occur at all, so there are more events the PDE performs poorly on.

In theory, neither memory availability (cost) nor the finiteness of the training data set is an issue. However, since the ultimate purpose of a video compression system is its application, these issues need to be dealt with.

The two issues discussed above suggest two different approaches to determining the escape codes and selecting the events to be escape coded. The two approaches will be discussed next. The first approach tries to solve the problem of the exceedingly long codewords, while the second one tackles the problem of the increased complexity and the increased memory requirement.

4.1 Limiting the Codeword Length

The second problem, i.e. the inability of a finite training set to predict the relative probabilities of all possible events, has as a consequence that the PDE scheme performs poorly when those events that were not predicted occur (the corresponding codewords are exceedingly long). Therefore, the codeword length of such unlikely events needs to be limited.

One approach is to consider the uniform-coder codeword length the benchmark which determines which events are to be escape coded and which ones are not. If a codeword for an event is longer than that benchmark length, the event is to be escape coded, and vice versa. The aggregate probability of all events that are to be escape coded determines the escape sequence. Since this process changes the codeword assignments for even the non-escape-coded events, there may be additional events in the new codebook with codewords longer than the benchmark. Therefore, the procedure is repeated until all non-escape-coded events have codewords shorter or equal to the benchmark length. Hence, the events are ultimately divided into two categories: the first category are the events that are entropy (Huffman) coded and that have variable length codewords not longer than the codewords in the uniform coder; the second are the events that are coded with the escape code followed by their corresponding uniform-coder codeword.

Although this scheme for determining the escape codes is iterative and therefore complex, determination of escape codes and codebooks happens only once, so the complexity is irrelevant.

4.2 Limiting the Codebook Size

The problems of the increased complexity and the increased memory requirement are most easily solved by limiting the codebook size. Once the number of entries in the codebooks (i.e. the size of the codebooks) is chosen, that many events are selected to be entropy encoded and all other events are escape coded. The events to be entropy coded could either be the few most likely events, or, to keep the implementation simple, the first few events in the ordering used.

The aggregate probability of events to be escape coded determines the escape code. This process changes the codeword assignments for even the events that are not escape coded. However, since the constrain is the number of entries in the codebooks and not the length of the codewords, the procedure does not need to be repeated. The code for the escape-coded events is comprised of the escape code followed by the appropriate uniform-coder codeword. This approach is much simpler than the approach discussed in Section 4.1 since it is a one-step procedure.

In the case that the first few events in a regular codebook are picked to be entropy coded the complexity of the implementation is smaller than if the few most probable events are picked to be entropy coded. If the first few events in this particular ordering are also the most likely ones (at least most of the time) only little performance is sacrificed as compared to always selecting the most likely events to be entropy coded.

Chapter 5

Results

This chapter presents the results of comparisons of the position-dependent encoding approach to the single-codebook separate and joint encoding approaches. First, though, is Section 5.1 which describes in detail the experiments performed.

5.1 Experimental Setup

The different encoding approaches were all implemented within the same CCDC-like compression system [4]. The compression algorithm used was based, as illustrated in Figure 5-1, on motion-compensated prediction and 8x8 spatial-block DCT. Adaptive quantization was performed on the basis of MPEG2-prescribed weighting tables [5]. An equal number of bits was assigned to both intra (I) and predicted (P) frames. No bidirectionally-predicted (B) frames were used. Video was progressively refreshed at the rate of 3 entire frames per second, as described in [4].

Bit Rate Considerations

The bit rate was nominally 0.35 bits/pixel. Constant bit rate was maintained using global buffer control. However, as will be obvious from Figures 5-2, 5-3, 5-4, and 5-5, the actual bit rate fluctuated around the 0.35 bits/pixel mark. There were two causes of these fluctuations.

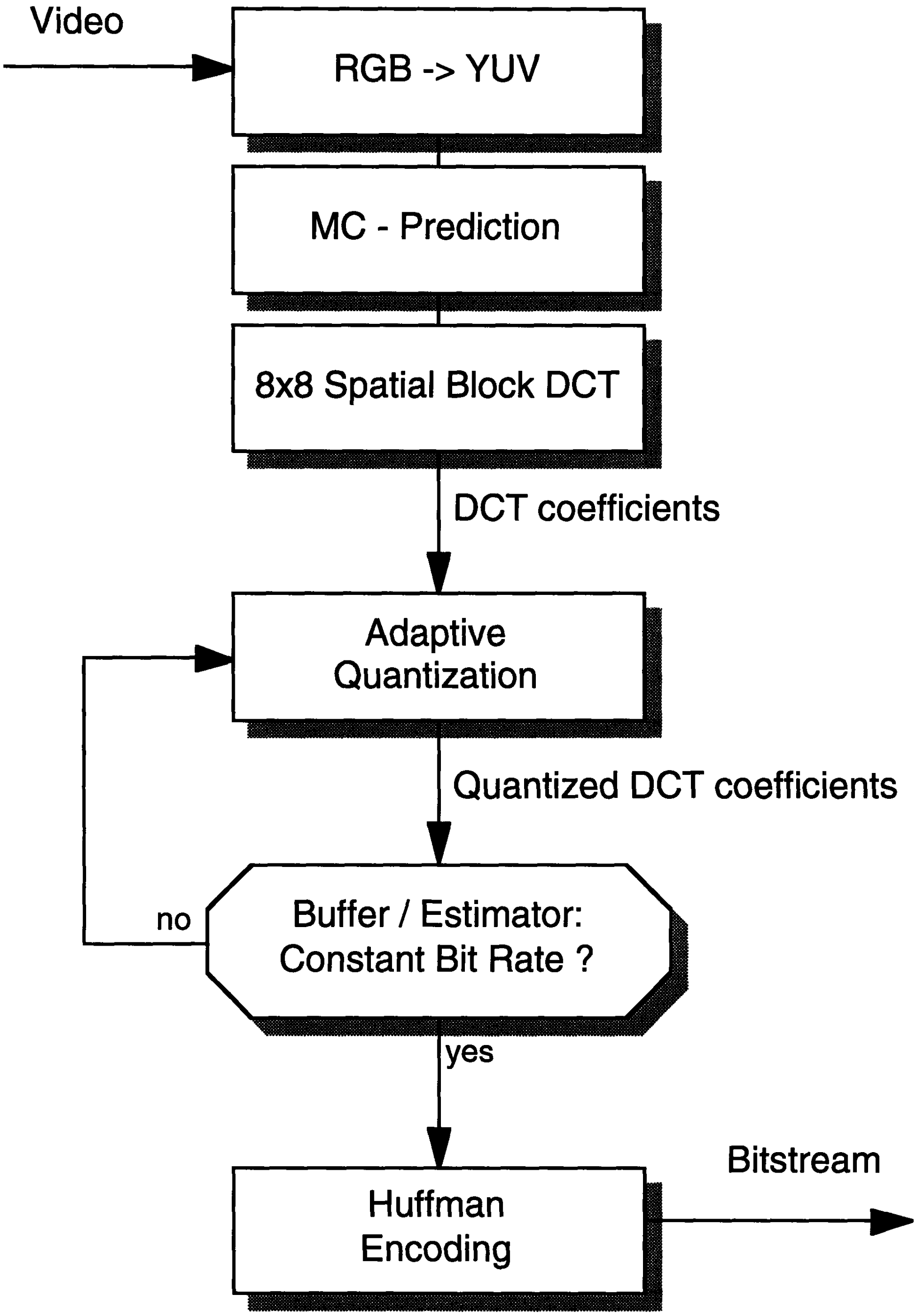


Figure 5-1: The compression system used for comparison of the encoding schemes. The Buffer/Estimator performs a quick estimate of the number of bits needed to encode a frame at the current quantization level.

First, the actual buffer control was achieved by quickly estimating the bit rate at a particular quantization level, instead of actually computing the exact number of bits needed to encode a frame at a different quantization level in each new iteration. The shortcut was taken because of the much improved efficiency in computing the quantized coefficients. The estimator used tended to underestimate the bit rate.

Second, the global (frame by frame) buffer control used in the compression system does not exploit the bit rate available as efficiently as the local (block by block) buffer control. Therefore, there tends to be a fraction of the bit rate left unused when the global control is used. This decreases the bit rate.

As can be seen from the bit rates achieved by using single-codebook separate or single-codebook joint encoding (Tables 5-2, 5-3, 5-4, 5-5), 9 out of 12 sequences had bit rates higher than the nominal 0.35 bits/pixel, while the other 3 have bit rates only slightly below the nominal bit rate. It can then be concluded that the first cause of the bit rate fluctuations dominated.

Comparison Basis

In order to pinpoint the differences in performance caused by the different runlength and amplitude encoding schemes, a common Intra DC coding scheme was chosen. In this particular implementation the Intra DC coefficient was encoded using a DPCM (Differential Pulse Code Modulation) scheme. This was necessary since the CCDC system [4] and the MPEG2 system [5] use different approaches to encoding the Intra DC coefficient, although both always encode it, as was mentioned in Chapter 2. Actually both systems use a version of the DPCM scheme, but with different implementation details. The basis of comparison is then the encoding of all coefficients other than Intra DC.

Sequences and Codebooks

The quantized DCT coefficient statistics were collected from 12 different video sequences, all of them progressively scanned. Six of these were 60 Hz camera video sequences obtained from Zenith (sequences 1-6), another four were 24 Hz film sequences obtained from Kodak and coded at 60 frames/second (sequences 7-10), while two were synthetically generated (zoom and pan) at MIT from stills obtained from Kodak (sequences 11 and 12). Other characteristics of the video sequences used are summarized in Table 5-1.

Table 5-1: Summary of the characteristics of the sequences used in training and testing.

source	60 Hz Video						24 Hz Film				Synthetic	
sequence	1	2	3	4	5	6	7	8	9	10	11	12
vertical resolution	720	720	720	720	720	720	704	880	880	880	512	720
horizontal resolution	1024	1024	1024	1024	1024	1024	1200	1200	1200	1200	512	1024
number of frames used	10	10	10	10	10	10	8	10	10	10	20	20

The Huffman codebooks were trained based on the set of runlength and amplitude statistics obtained as a weighted sum of the individual statistics of the twelve sequences. The tests were performed on the same twelve sequences. The results are summarized in the following sections.

5.2 The Ultimate Position-Dependent Encoding

If the position-dependent encoding approach is taken to its extreme, every DCT coefficient should have its own codebook. Theoretically, having the Huffman codebooks trained to the most local of the statistics, should produce coding performance superior to any other distribution of codebooks. Given the finite training set, the ultimate PDE should perform the best, at least, on average.

A total of 254 codebooks were used for runlength coding: 63 codebooks for coding of the Intra luminance component (Y) runlengths, 63 codebooks for Intra chrominance components (UV) runlengths, 64 for Inter Y runlengths, and 64 for Inter UV runlengths. Similarly, 254 codebooks were used for amplitude coding: 63 codebooks for each Intra Y amplitudes and Intra UV amplitudes, and 64 codebooks for each Inter Y and Inter UV amplitudes. Since Intra DC coefficients are not coded using the PDE, as noted in Section 5.1, only 63 codebooks are used for intra blocks, as opposed to 64 used for inter blocks.

The performance results are summarized in Table 5-2 and Figure 5-2. The bit/pixel rates shown are the ratio of the total number of bits used for encoding a particular video sequence to the total number of pixels in that sequence (i.e. the number of frames multiplied by the horizontal and vertical resolutions). The performance of the PDE is presented in terms of the percentage decrease of the bit/pixel rate of the PDE over the two single-codebook encoding schemes. All tests were, of course, done on the exact same sets of quantized DCT coefficients. Figures 5-2 (a)-(i) graph the bit/pixel rates for intra and inter block runlengths and amplitudes, both individually and for various totals.

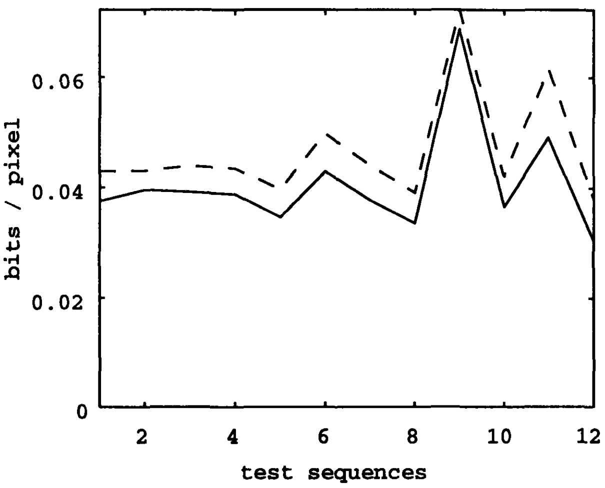
As the plots and table indicate, the largest improvement was achieved in coding the intra-block runlengths, while the smallest improvement was achieved in coding the inter-block amplitudes. In general, runlength coding yields a larger decrease in the bit rate than amplitude coding. Since runlengths occupy more than 60% of the total bit rate, this is advantageous. The coding of intra blocks yields a larger improvement then does the coding of inter blocks. However, since intra blocks occupy less than 30% of the bit rate, this does not produce any advantage.

The ultimate PDE performs better then either of the single-codebook schemes for all twelve sequences. However, as can be seen from Figure 5-2 (i), the differences between the PDE and the joint encoding vary widely across sequences. The average total decrease in the bit rate achieved with the PDE when compared to the single-codebook separate encoding is 6.6%. The average total decrease in the bit rate achieved with the PDE when compared to the single-codebook joint encoding is 6.2%, only slightly lower than the 6.6% decrease.

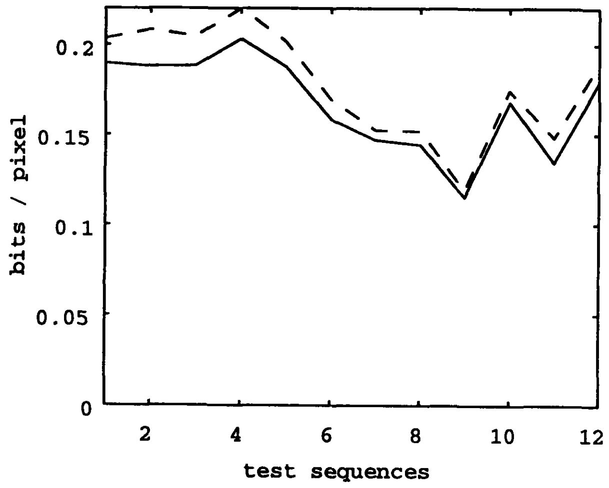
Table 5-2: The Ultimate Position-Dependent Encoding vs. single-codebook separate and joint encoding. The Ultimate PDE uses 254 runlength and 254 amplitude codebooks.

		source	60 Hz Video						24 Hz Film				Synthetic		
		sequence	1	2	3	4	5	6	7	8	9	10	11	12	average
Runlengths	Intra	separate (bits/pixel)	.0429	.0430	.0439	.0434	.0397	.0497	.0440	.0390	.0723	.0421	.0616	.0382	.0467
		PDE (bits/pixel)	.0376	.0395	.0392	.0387	.0346	.0429	.0377	.0335	.0687	.0366	.0491	.0306	.0407
		improvement (%)	**14.3**	**8.7**	**12.1**	**12.1**	**14.6**	**16.0**	**16.9**	**16.4**	**5.2**	**15.0**	**25.4**	**24.9**	**14.6**
	Inter	separate (bits/pixel)	.2035	.2083	.2047	.2201	.2016	.1698	.1524	.1519	.1189	.1747	.1482	.1894	.1786
		PDE (bits/pixel)	.1897	.1882	.1884	.2033	.1879	.1585	.1469	.1443	.1152	.1681	.1343	.1791	.1670
		improvement (%)	**7.3**	**10.7**	**8.7**	**8.3**	**7.3**	**7.1**	**3.7**	**5.3**	**3.3**	**3.9**	**10.3**	**5.8**	**7.0**
	Total	separate (bits/pixel)	.2465	.2513	.2486	.2636	.2413	.2195	.1964	.1909	.1912	.2167	.2098	.2276	.2253
		PDE (bits/pixel)	.2273	.2277	.2275	.2420	.2225	.2014	.1846	.1778	.1839	.2047	.1834	.2097	.2077
		improvement (%)	**8.4**	**10.3**	**9.2**	**8.9**	**8.4**	**9.0**	**6.4**	**7.4**	**4.0**	**5.9**	**14.4**	**8.5**	**8.5**
Amplitudes	Intra	separate (bits/pixel)	.0519	.0469	.0491	.0469	.0542	.0687	.0633	.0582	.0818	.0579	.0906	.0607	.0608
		PDE (bits/pixel)	.0490	.0439	.0463	.0447	.0504	.0627	.0572	.0537	.0774	.0535	.0851	.0573	.0568
		improvement (%)	**6.0**	**6.6**	**6.0**	**4.8**	**7.6**	**9.7**	**10.7**	**8.4**	**5.7**	**8.2**	**6.5**	**5.9**	**7.2**
	Inter	separate (bits/pixel)	.0721	.0697	.0682	.0716	.0739	.0896	.0827	.0910	.0737	.0892	.0788	.0825	.0786
		PDE (bits/pixel)	.0718	.0693	.0681	.0714	.0736	.0876	.0813	.0892	.0727	.0881	.0771	.0816	.0776
		improvement (%)	**0.4**	**0.7**	**0.2**	**0.2**	**0.5**	**2.4**	**1.6**	**2.1**	**1.3**	**1.3**	**2.2**	**1.2**	**1.2**
	Total	separate (bits/pixel)	.1240	.1166	.1173	.1184	.1282	.1584	.1460	.1492	.1555	.1471	.1694	.1432	.1394
		PDE (bits/pixel)	.1208	.1132	.1143	.1161	.1240	.1502	.1385	.1428	.1501	.1415	.1622	.1389	.1344
		improvement (%)	**2.7**	**3.0**	**2.6**	**2.0**	**3.4**	**5.4**	**5.4**	**4.5**	**3.6**	**3.9**	**4.4**	**3.1**	**3.8**
Total	Intra	separate (bits/pixel)	.0949	.0898	.0930	.0903	.0939	.1185	.1073	.0972	.1542	.0999	.1522	.0989	.1075
		PDE (bits/pixel)	.0866	.0835	.0854	.0835	.0850	.1056	.0948	.0872	.1462	.0900	.1342	.0879	.0975
		improvement (%)	**9.6**	**7.6**	**8.8**	**8.2**	**10.4**	**12.2**	**13.2**	**11.5**	**5.5**	**11.0**	**13.4**	**12.5**	**10.3**
	Inter	separate (bits/pixel)	.2756	.2780	.2729	.2917	.2755	.2594	.2351	.2429	.1926	.2639	.2270	.2719	.2572
		PDE (bits/pixel)	.2615	.2575	.2564	.2747	.2615	.2461	.2283	.2334	.1879	.2562	.2114	.2606	.2446
		improvement (%)	**5.4**	**8.0**	**6.4**	**6.2**	**5.4**	**5.4**	**3.0**	**4.1**	**2.5**	**3.0**	**7.4**	**4.3**	**5.1**
	Total	separate (bits/pixel)	.3705	.3679	.3659	.3820	.3694	.3779	.3424	.3401	.3468	.3638	.3791	.3708	.3647
		PDE (bits/pixel)	.3481	.3409	.3419	.3582	.3465	.3516	.3231	.3207	.3340	.3463	.3456	.3485	.3421
		improvement (%)	**6.4**	**7.9**	**7.0**	**6.7**	**6.6**	**7.5**	**6.0**	**6.1**	**3.8**	**5.1**	**9.7**	**6.4**	**6.6**
		joint (bits/pixel)	.3806	.4001	.3775	.4012	.3742	.3618	.3273	.3223	.3355	.3509	.3713	.3591	.3635
		PDE vs. joint (%)	**9.3**	**17.4**	**10.4**	**12.0**	**8.0**	**2.9**	**1.3**	**0.5**	**0.4**	**1.3**	**7.4**	**3.0**	**6.2**

(a) Intra Runlengths



(b) Inter Runlengths

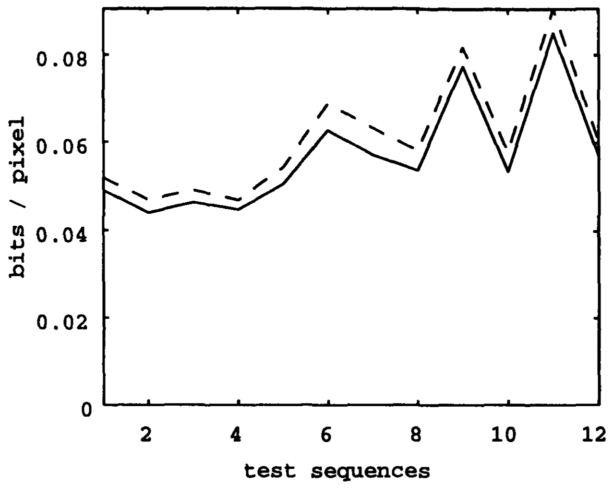


_ _ _ single-codebook separate encoding

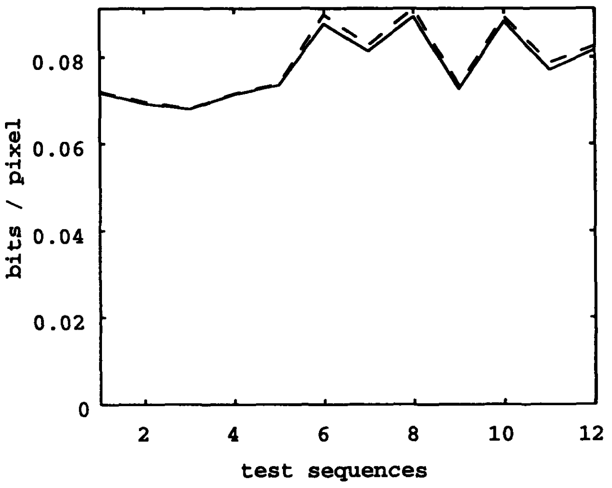
____ position-dependent encoding

Figure 5-2: The Ultimate PDE vs. single-codebook separate and joint encoding: (a) Intra Runlengths, (b) Inter Runlengths.

(c) Intra Amplitudes



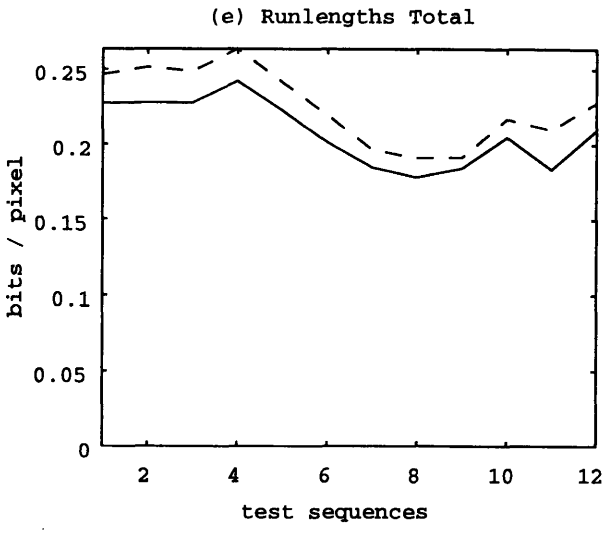
(d) Inter Amplitudes



- - - single-codebook separate encoding

—— position-dependent encoding

Figure 5-2: (continued) The Ultimate PDE vs. single-codebook separate and joint encoding: (c) Intra Amplitudes, (d) Inter Amplitudes.



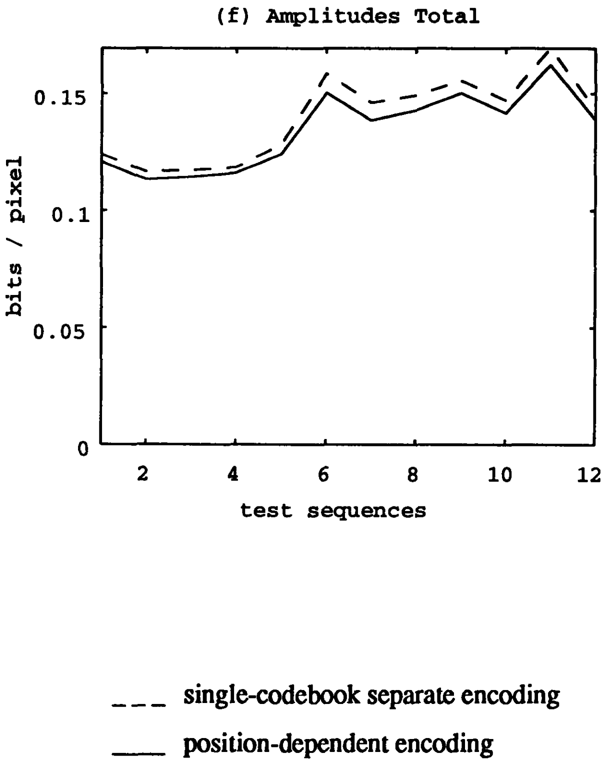
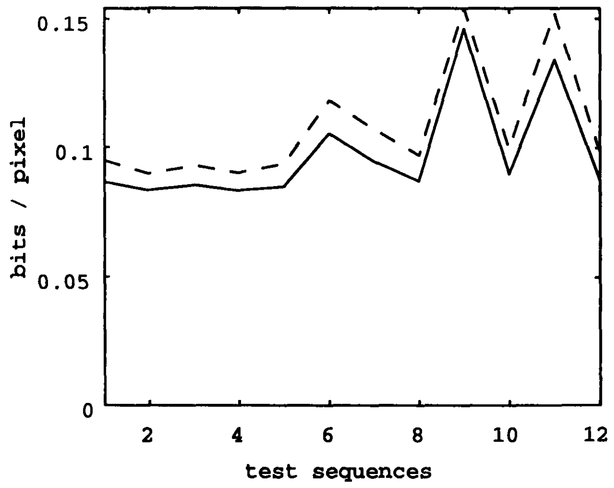
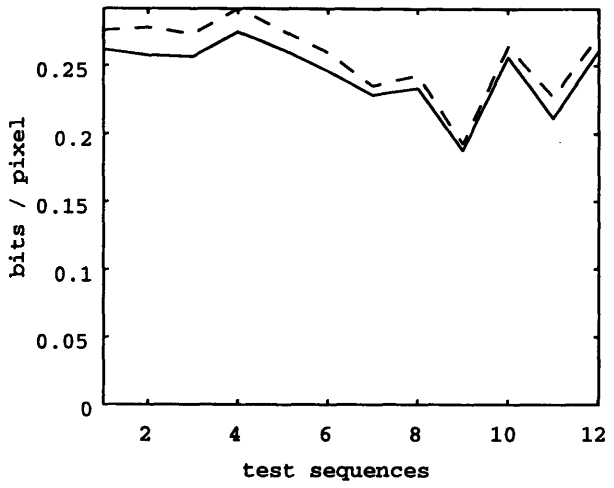


Figure 5-2: (continued) The Ultimate PDE vs. single-codebook separate and joint encoding: (e) Runlengths Total, (f) Amplitudes Total.

(g) Intra Total



(h) Inter Total



_ _ _ single-codebook separate encoding

____ position-dependent encoding

Figure 5-2: (continued) The Ultimate PDE vs. single-codebook separate and joint encoding: (g) Intra Total, (h) Inter Total.

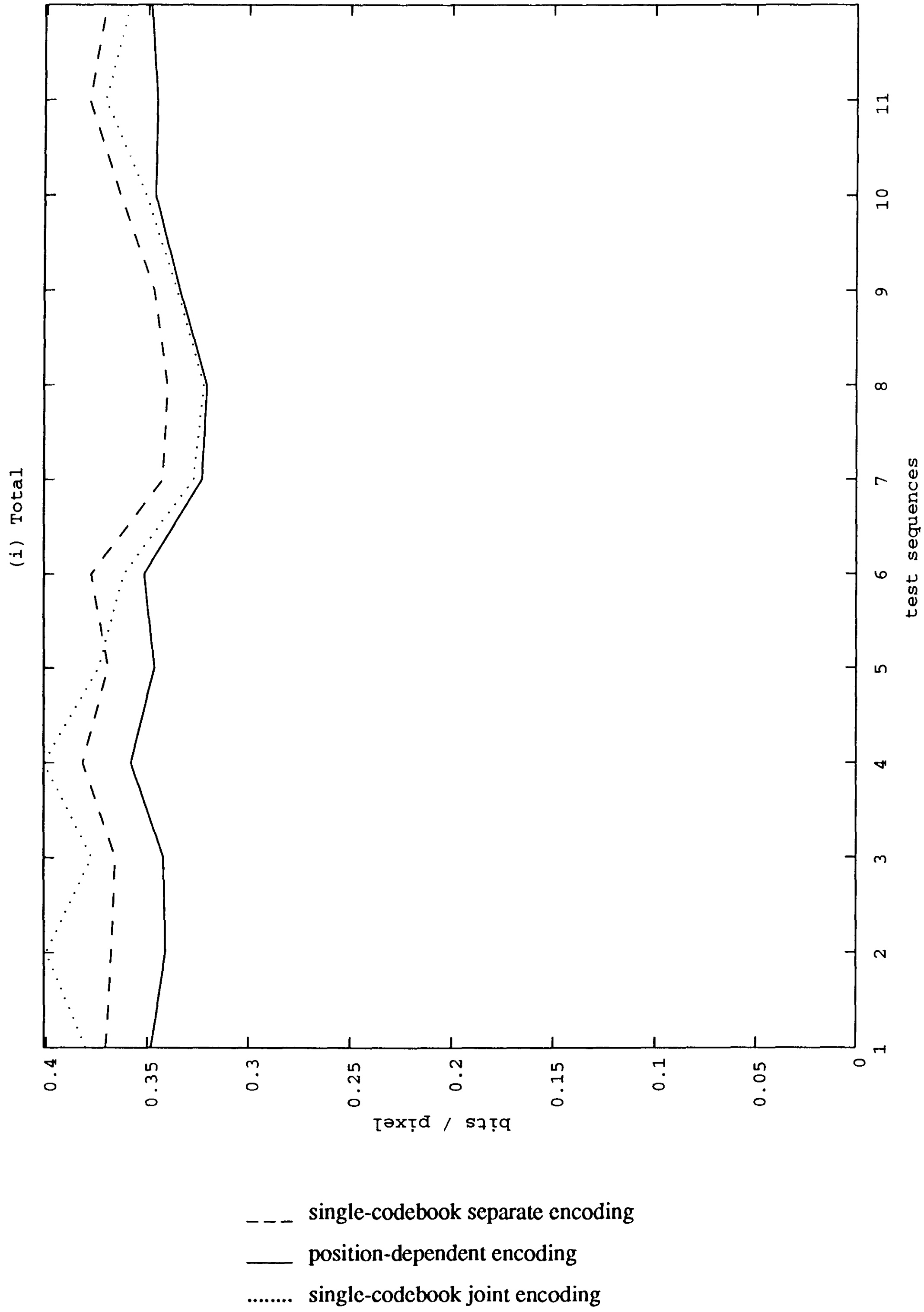


Figure 5-2: (continued) The Ultimate PDE vs. single-codebook separate and joint encoding: (i) Total.

5.3 The Proposed Position-Dependent Encoding

A total of 508 codebooks was used for the ultimate PDE scheme discussed in Section 5.2. Considering the memory requirement, the scheme with this many codebooks is absolutely impractical. Therefore, it is imperative to decrease the number of codebooks by having the coefficients share codebooks.

A decrease in the number of codebooks also means decrease in the coding benefits of the PDE. But the two are not necessarily commensurate. As the results presented in this section show, it is still possible to get most of the performance gain with significantly fewer codebooks.

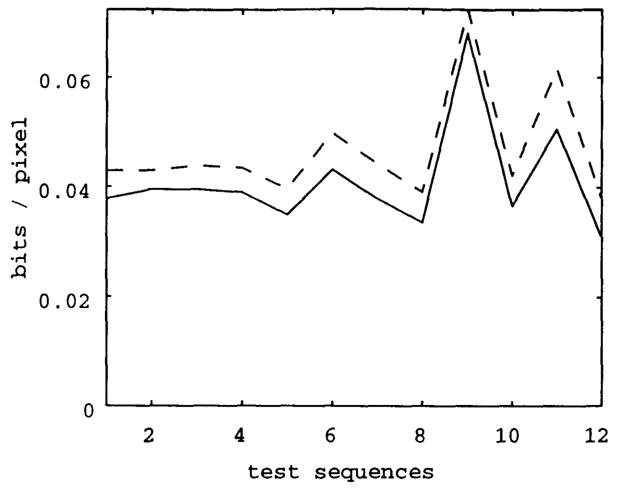
The proposed PDE scheme, also described in [7], uses 94 codebooks for runlength coding: 31 for Intra Y runlengths, 15 for Intra UV, 32 for Inter Y, and 16 for Inter UV. A total of 14 amplitude codebooks was distributed in the following manner: 3 for each Intra Y and Intra UV, and 4 for each Inter Y and Inter UV. The exact pattern for the codebook selection is included in Appendix A. Since improving runlength coding is of greater importance than improving amplitude coding (as noted in Section 5.2), many more codebooks of the total 108 have been devoted to runlengths than to amplitudes. An (almost) equal number of codebooks was assigned to both the intra and inter blocks. While the intra blocks yield larger relative improvement with the same number of codebooks than the inter blocks, the inter blocks are more important since they occupy more bit rate. Each intra block was assigned one codebook less than the corresponding inter block because, while Inter DC had its own codebook, Intra DC was not coded using the PDE.

The results for the proposed PDE scheme are summarized in Table 5-3 and Figures 5-3 (a)-(i). The proposed PDE performs better than both the single-codebook separate encoding (by 6.1%), and the single-codebook joint encoding (by 5.8%). As can be seen by comparing these results to the ones of Section 5.2, the 108-codebooks PDE actually outperforms the 508-codebooks PDE on 5 out of 12 sequences. Also, note that the performance gain in coding the amplitudes is virtually unchanged: 3.8% improvement with the 508-codebook scheme, and 3.7% improvement with the 108-codebook scheme. On average the ultimate PDE still performs better by a difference of 0.5%. However, the sacrifice of a half of a percent out of over 6% is minor compared to the reduction in the number of codebooks used.

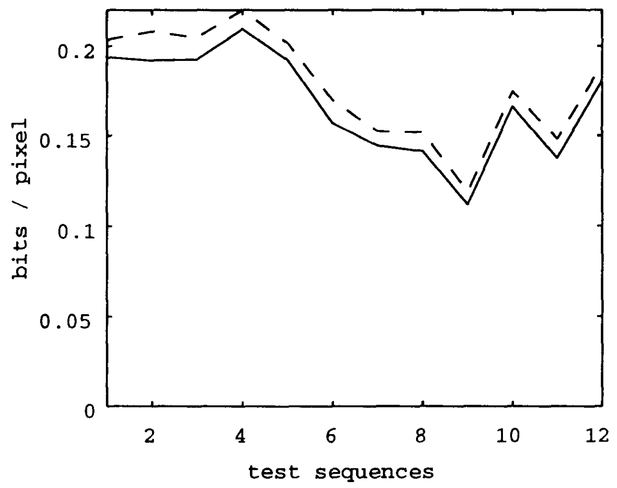
Table 5-3: The Proposed Position-Dependent Encoding vs. single-codebook separate and joint encoding.
The Proposed PDE uses 94 runlength and 14 amplitude codebooks.

		source	60 Hz Video						24 Hz Film				Synthetic		
		sequence	1	2	3	4	5	6	7	8	9	10	11	12	average
Runlengths	Intra	separate (bits/pixel)	.0429	.0430	.0439	.0434	.0397	.0497	.0440	.0390	.0723	.0421	.0616	.0382	.0467
		PDE (bits/pixel)	.0378	.0396	.0396	.0390	.0349	.0430	.0378	.0335	.0680	.0365	.0506	.0312	.0410
		improvement (%)	**13.6**	**8.6**	**10.9**	**11.2**	**13.5**	**15.6**	**16.5**	**16.5**	**6.3**	**15.1**	**21.7**	**22.6**	**13.9**
	Inter	separate (bits/pixel)	.2035	.2083	.2047	.2201	.2016	.1698	.1524	.1519	.1189	.1747	.1482	.1894	.1786
		PDE (bits/pixel)	.1937	.1920	.1926	.2095	.1922	.1570	.1444	.1414	.1118	.1662	.1373	.1807	.1682
		improvement (%)	**5.1**	**8.5**	**6.3**	**5.1**	**4.9**	**8.1**	**5.6**	**7.4**	**6.4**	**5.1**	**7.9**	**4.8**	**6.2**
	Total	separate (bits/pixel)	.2465	.2513	.2486	.2636	.2413	.2195	.1964	.1909	.1912	.2167	.2098	.2276	.2253
		PDE (bits/pixel)	.2315	.2315	.2323	.2485	.2272	.2001	.1822	.1749	.1798	.2027	.1879	.2118	.2092
		improvement (%)	**6.4**	**8.5**	**7.1**	**6.0**	**6.2**	**9.7**	**7.8**	**9.1**	**6.4**	**6.9**	**11.6**	**7.5**	**7.7**
Amplitudes	Intra	separate (bits/pixel)	.0519	.0469	.0491	.0469	.0542	.0687	.0633	.0582	.0818	.0579	.0906	.0607	.0608
		PDE (bits/pixel)	.0490	.0439	.0462	.0447	.0504	.0626	.0572	.0536	.0774	.0535	.0854	.0577	.0568
		improvement (%)	**6.0**	**6.7**	**6.1**	**4.9**	**7.6**	**9.9**	**10.7**	**8.5**	**5.7**	**8.2**	**6.2**	**5.2**	**7.1**
	Inter	separate (bits/pixel)	.0721	.0697	.0682	.0716	.0739	.0896	.0827	.0910	.0737	.0892	.0788	.0825	.0786
		PDE (bits/pixel)	.0718	.0693	.0681	.0714	.0736	.0875	.0813	.0891	.0727	.0881	.0774	.0816	.0776
		improvement (%)	**0.4**	**0.7**	**0.2**	**0.2**	**0.5**	**2.4**	**1.6**	**2.1**	**1.3**	**1.3**	**1.8**	**1.2**	**1.2**
	Total	separate (bits/pixel)	.1240	.1166	.1173	.1184	.1282	.1584	.1460	.1492	.1555	.1471	.1694	.1432	.1394
		PDE (bits/pixel)	.1208	.1132	.1143	.1161	.1239	.1501	.1385	.1428	.1501	.1415	.1627	.1392	.1344
		improvement (%)	**2.7**	**3.0**	**2.6**	**2.0**	**3.4**	**5.5**	**5.4**	**4.5**	**3.6**	**3.9**	**4.1**	**2.8**	**3.7**
Total	Intra	separate (bits/pixel)	.0949	.0898	.0930	.0903	.0939	.1185	.1073	.0972	.1542	.0999	.1522	.0989	.1075
		PDE (bits/pixel)	.0868	.0835	.0858	.0837	.0853	.1056	.0950	.0871	.0145	.0900	.1359	.0888	.0978
		improvement (%)	**9.3**	**7.6**	**8.3**	**7.8**	**10.0**	**12.2**	**13.0**	**11.6**	**6.0**	**11.0**	**11.9**	**11.3**	**10.0**
	Inter	separate (bits/pixel)	.2756	.2780	.2729	.2917	.2755	.2594	.2351	.2429	.1926	.2639	.2270	.2719	.2572
		PDE (bits/pixel)	.2655	.2613	.2607	.2809	.2658	.2445	.2257	.2305	.1845	.2542	.2147	.2622	.2459
		improvement (%)	**3.8**	**6.4**	**4.7**	**3.9**	**3.7**	**6.1**	**4.1**	**5.4**	**4.4**	**3.8**	**5.7**	**3.7**	**4.6**
	Total	separate (bits/pixel)	.3705	.3679	.3659	.3820	.3694	.3779	.3424	.3401	.3468	.3638	.3791	.3708	.3647
		PDE (bits/pixel)	.3523	.3447	.3465	.3646	.3511	.3502	.3207	.3177	.3299	.3443	.3506	.3511	.3436
		improvement (%)	**5.2**	**6.7**	**5.6**	**4.8**	**5.2**	**7.9**	**6.8**	**7.1**	**5.1**	**5.7**	**8.1**	**5.6**	**6.1**
		joint (bits/pixel)	.3806	.4001	.3775	.4012	.3742	.3618	.3273	.3223	.3355	.3509	.3713	.3591	.3635
		PDE vs. joint (%)	**8.0**	**16.1**	**8.9**	**10.0**	**6.6**	**3.3**	**2.1**	**1.5**	**1.7**	**1.9**	**5.9**	**2.3**	**5.8**

(a) Intra Runlengths



(b) Inter Runlengths

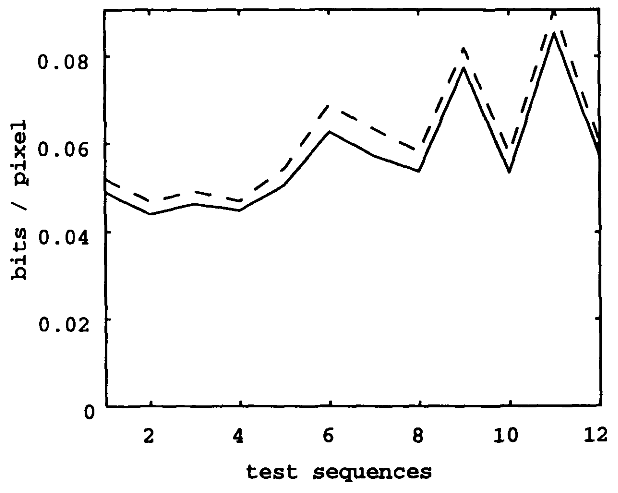


_ _ _ single-codebook separate encoding

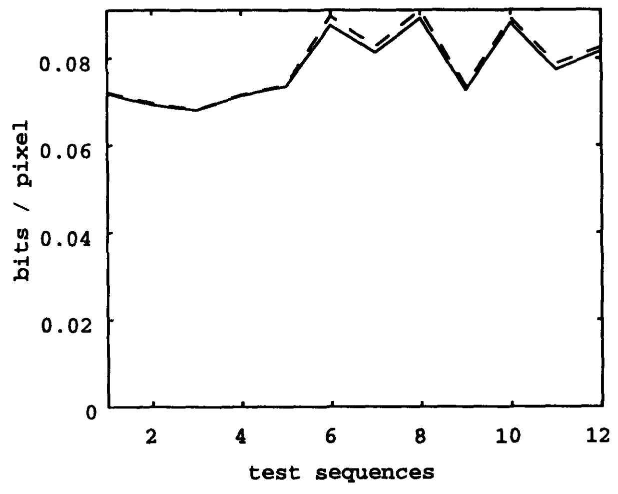
____ position-dependent encoding

Figure 5-3: The Proposed PDE vs. single-codebook separate and joint encoding: (a) Intra Runlengths, (b) Inter Runlengths.

(c) Intra Amplitudes



(d) Inter Amplitudes

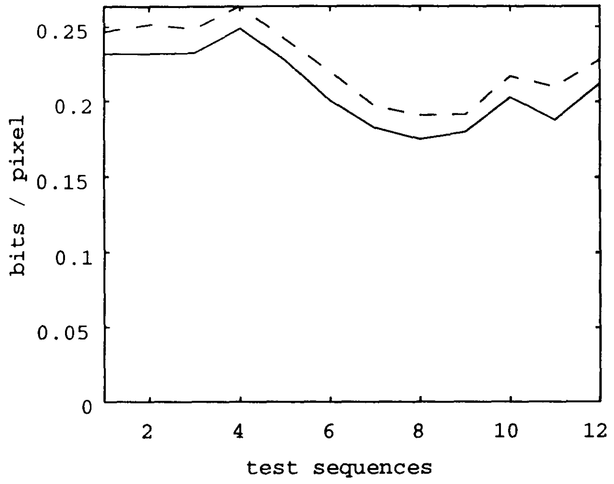


_ _ _ single-codebook separate encoding

____ position-dependent encoding

Figure 5-3: (continued) The Proposed PDE vs. single-codebook separate and joint encoding: (c) Intra Amplitudes, (d) Inter Amplitudes.

(e) Runlengths Total



(f) Amplitudes Total

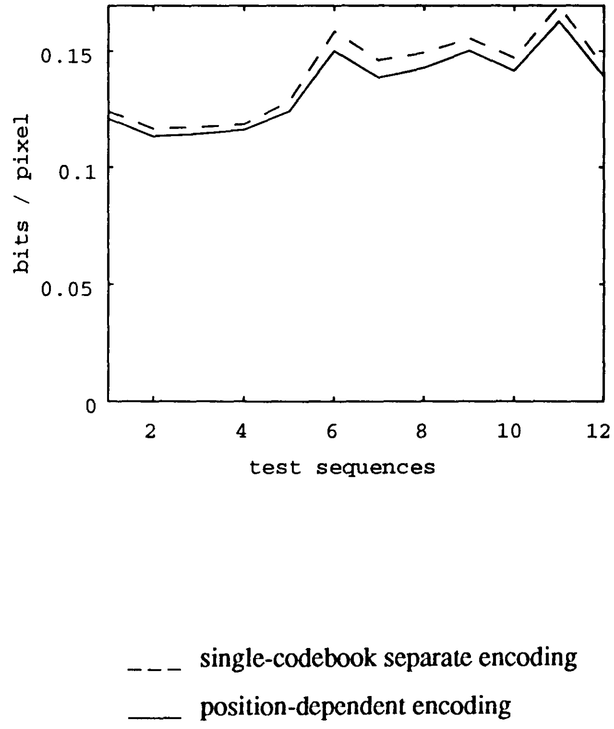
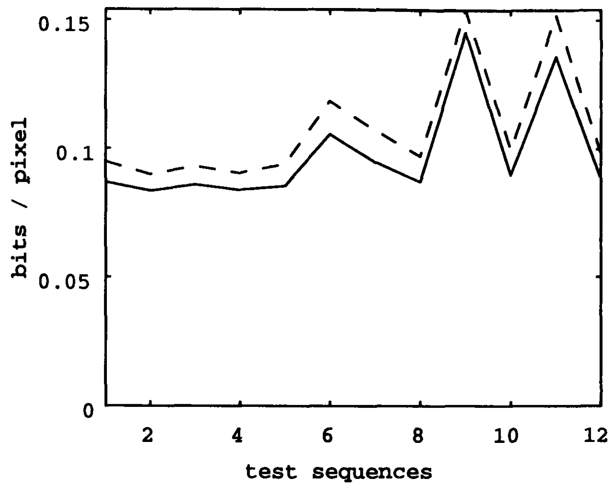
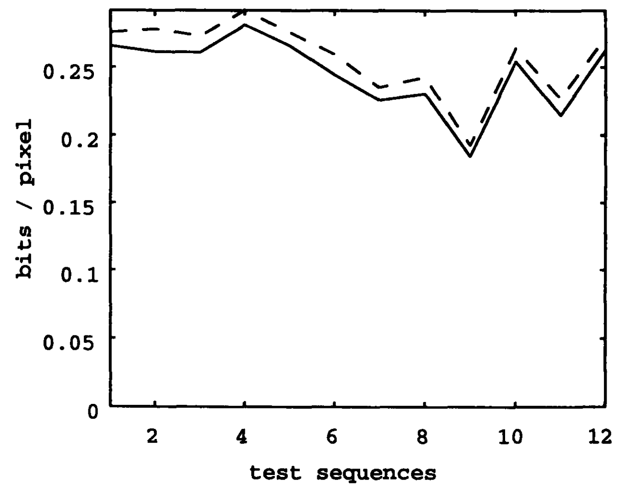


Figure 5-3: (continued) The Proposed PDE vs. single-codebook separate and joint encoding: (e) Runlengths Total, (f) Amplitudes Total.

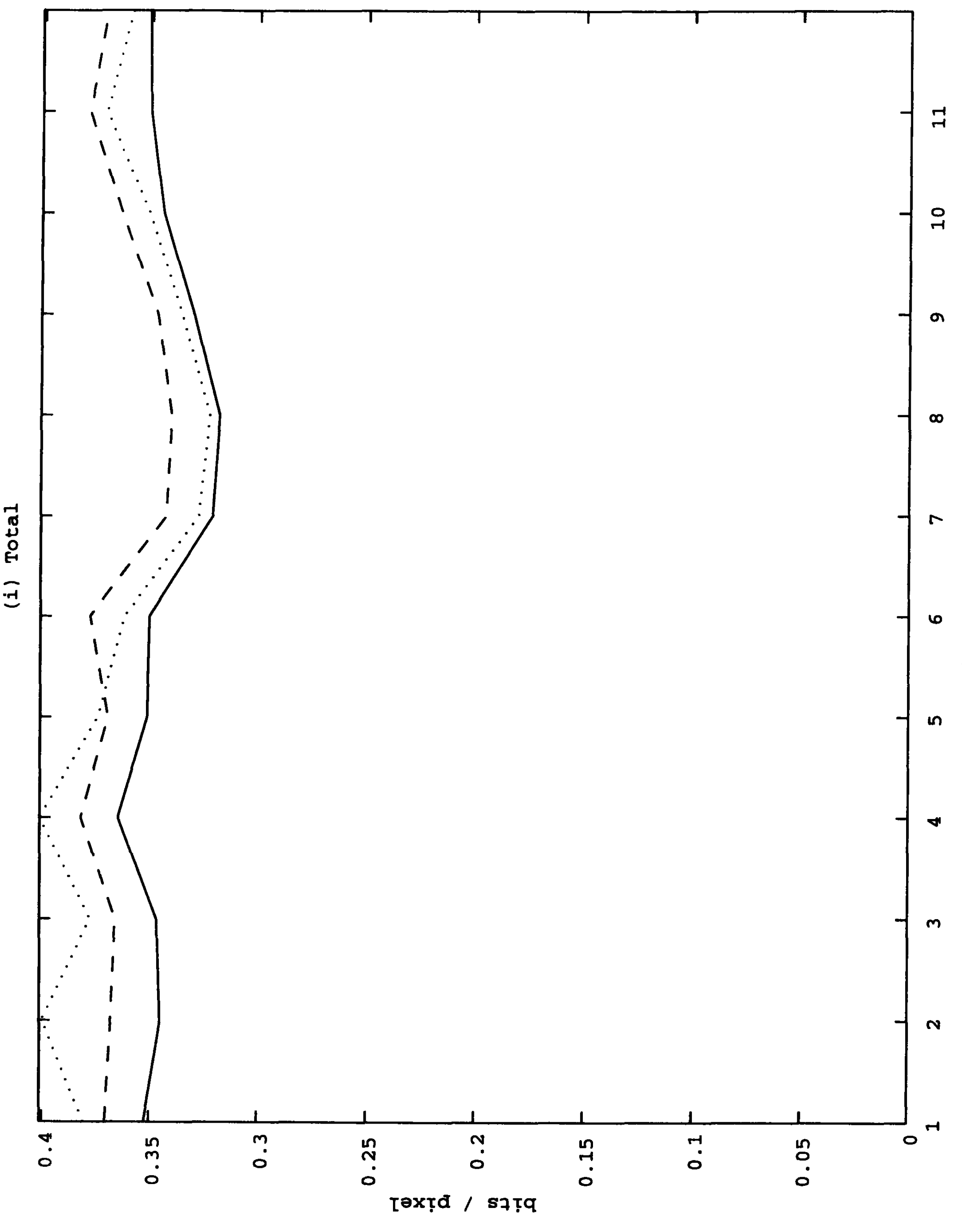

(g) Intra Total


(h) Inter Total

_ _ _ single-codebook separate encoding

____ position-dependent encoding

Figure 5-3: (continued) The Proposed PDE vs. single-codebook separate and joint encoding: (g) Intra Total, (h) Inter Total.



_ _ _ single-codebook separate encoding

____ position-dependent encoding

.... single-codebook joint encoding

Figure 5-3: (continued) The Proposed PDE vs. single-codebook separate and joint encoding: (i) Total.

5.4 The Position-Dependent Encoding with Escape Codes Limiting the Codeword Length

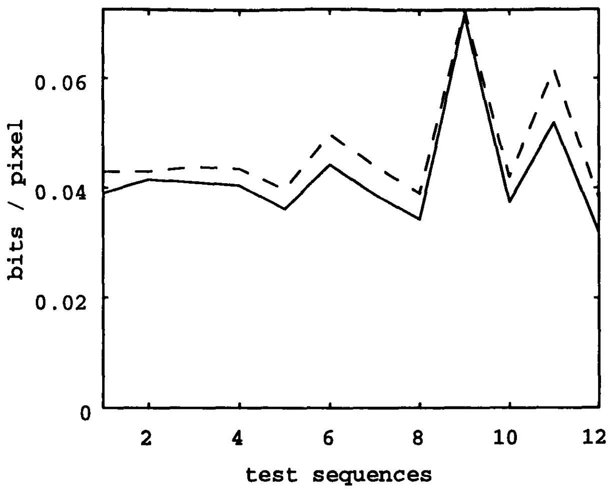
The finite training set cannot predict the probabilities of all possible runlength and amplitude events. The events that did not occur within the training set then have exceedingly long codewords in the Huffman codebooks. Those events could occur in new sequences, however, and in that case the PDE would perform poorly. In order to prevent that from happening, codeword length needs to be limited by introducing the escape codes. In this section escape codes were created using the approach described in Section 4-1. A sample codebook with escape codes is included in Appendix B.

The performance results for the 108-codebooks PDE with escape codes chosen in the above manner are summarized in Table 5-4 and Figures 5-4 (a)-(i). As can be seen from the material included, the performance improvement of the proposed PDE is diminished by a difference of about 3% (3.8% improvement over separate encoding and 2.8% improvement over joint encoding). The gain over the single-codebook encoding approaches has diminished to half of its original value. Although, the PDE scheme with escape codes still performs better, on average, than either of the other two approaches, it actually performs worse than single-codebook joint encoding on 5 out of 12 sequences. Henceforth, there is not much motivation to use the PDE scheme with escape codes limiting the codeword length over the single-codebook approaches, especially the joint encoding scheme.

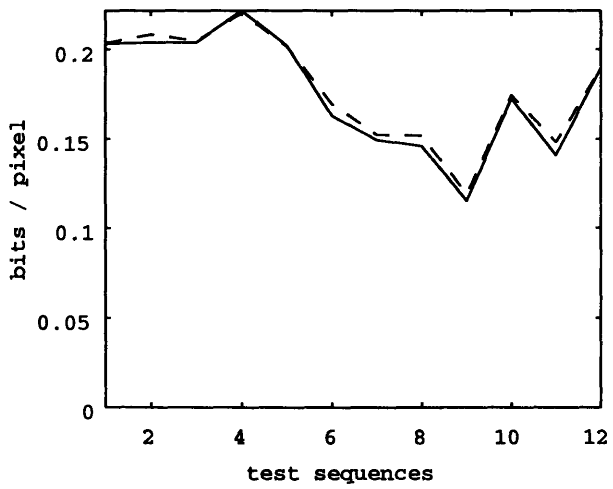
Table 5-4: The Position-Dependent Encoding with Escape Codes obtained by limiting the codeword length vs. single-codebook separate and joint encoding. The PDE uses 94 runlength and 14 amplitude codebooks.

		source	60 Hz Video						24 Hz Film				Synthetic		
		sequence	1	2	3	4	5	6	7	8	9	10	11	12	average
Runlengths	Intra	separate (bits/pixel)	.0429	.0430	.0439	.0434	.0397	.0497	.0440	.0390	.0723	.0421	.0616	.0382	.0467
		PDE (bits/pixel)	.0390	.0415	.0410	.0404	.0360	.0441	.0388	.0343	.0714	.0375	.0520	.0320	.0423
		improvement (%)	**10.2**	**3.6**	**7.0**	**7.6**	**10.1**	**12.7**	**13.6**	**14.0**	**1.2**	**12.2**	**18.4**	**19.3**	**10.2**
	Inter	separate (bits/pixel)	.2035	.2083	.2047	.2201	.2016	.1698	.1524	.1519	.1189	.1747	.1482	.1894	.1786
		PDE (bits/pixel)	.2030	.2038	.2039	.2216	.2020	.1631	.1495	.1459	.1151	.1728	.1408	.1895	.1759
		improvement (%)	**0.2**	**2.2**	**0.4**	**-0.7**	**-0.2**	**4.1**	**2.0**	**4.1**	**3.3**	**1.1**	**5.3**	**-0.1**	**1.5**
	Total	separate (bits/pixel)	.2465	.2513	.2486	.2636	.2413	.2195	.1964	.1909	.1912	.2167	.2098	.2276	.2253
		PDE (bits/pixel)	.2420	.2453	.2449	.2620	.2381	.2072	.1882	.1802	.1865	.2103	.1927	.2215	.2183
		improvement (%)	**1.8**	**2.4**	**1.5**	**0.6**	**1.3**	**6.0**	**4.4**	**5.9**	**2.5**	**3.1**	**8.8**	**2.7**	**3.2**
Amplitudes	Intra	separate (bits/pixel)	.0519	.0469	.0491	.0469	.0542	.0687	.0633	.0582	.0818	.0579	.0906	.0607	.0608
		PDE (bits/pixel)	.0492	.0440	.0464	.0448	.0508	.0626	.0576	.0539	.0777	.0540	.0870	.0593	.0573
		improvement (%)	**5.4**	**6.5**	**5.9**	**4.6**	**6.8**	**9.7**	**9.9**	**8.0**	**5.3**	**7.1**	**4.2**	**2.3**	**6.2**
	Inter	separate (bits/pixel)	.0721	.0697	.0682	.0716	.0739	.0896	.0827	.0910	.0737	.0892	.0788	.0825	.0786
		PDE (bits/pixel)	.0718	.0694	.0681	.0714	.0737	.0876	.0820	.0898	.0739	.0889	.0788	.0818	.0781
		improvement (%)	**0.4**	**0.5**	**0.2**	**0.2**	**0.4**	**2.4**	**0.9**	**1.4**	**-0.2**	**0.4**	**0.0**	**0.9**	**0.6**
	Total	separate (bits/pixel)	.1240	.1166	.1173	.1184	.1282	.1584	.1460	.1492	.1555	.1471	.1694	.1432	.1394
		PDE (bits/pixel)	.1211	.1134	.1144	.1162	.1244	.1502	.1395	.1437	.1516	.1429	.1658	.1411	.1354
		improvement (%)	**2.4**	**2.8**	**2.5**	**1.9**	**3.0**	**5.4**	**4.6**	**3.9**	**2.6**	**2.9**	**2.2**	**1.4**	**3.0**
Total	Intra	separate (bits/pixel)	.0949	.0898	.0930	.0903	.0939	.1185	.1073	.0972	.1542	.0999	.1522	.0989	.1075
		PDE (bits/pixel)	.0882	.0855	.0874	.0852	.0868	.1068	.0963	.0882	.1491	.0915	.1389	.0913	.0996
		improvement (%)	**7.5**	**5.1**	**6.4**	**6.0**	**8.2**	**11.0**	**11.4**	**10.3**	**3.4**	**9.2**	**9.5**	**8.2**	**8.0**
	Inter	separate (bits/pixel)	.2756	.2780	.2729	.2917	.2755	.2594	.2351	.2429	.1926	.2639	.2270	.2719	.2572
		PDE (bits/pixel)	.2749	.2732	.2720	.2930	.2757	.2506	.2314	.2357	.1890	.2617	.2196	.2713	.2540
		improvement (%)	**0.3**	**1.8**	**0.4**	**-0.4**	**-0.1**	**3.5**	**1.6**	**3.1**	**1.9**	**0.8**	**3.4**	**0.2**	**1.3**
	Total	separate (bits/pixel)	.3705	.3679	.3659	.3820	.3694	.3779	.3424	.3401	.3468	.3638	.3791	.3708	.3647
		PDE (bits/pixel)	.3631	.3587	.3593	.3782	.3625	.3574	.3278	.3239	.3381	.3532	.3585	.3627	.3536
		improvement (%)	**2.0**	**2.6**	**1.8**	**1.0**	**1.9**	**5.7**	**4.5**	**5.0**	**2.6**	**3.0**	**5.8**	**2.2**	**3.1**
		joint (bits/pixel)	.3806	.4001	.3775	.4012	.3742	.3618	.3273	.3223	.3355	.3509	.3713	.3591	.3635
		PDE vs. joint (%)	**4.8**	**11.5**	**5.1**	**6.1**	**3.2**	**1.2**	**-0.1**	**-0.5**	**-0.8**	**-0.6**	**3.6**	**-1.0**	**2.8**

(a) Intra Runlengths



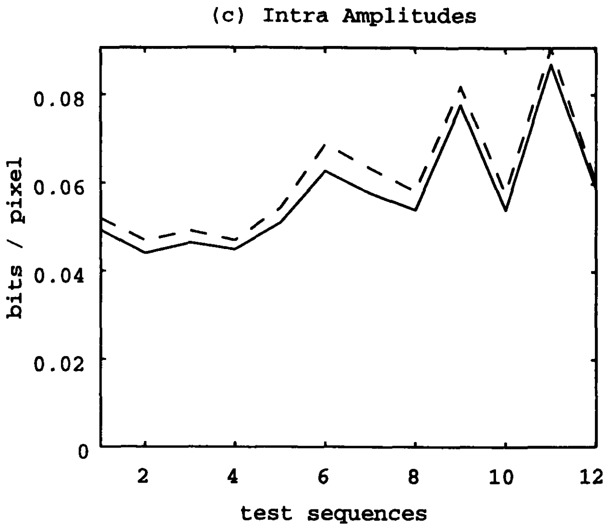
(b) Inter Runlengths

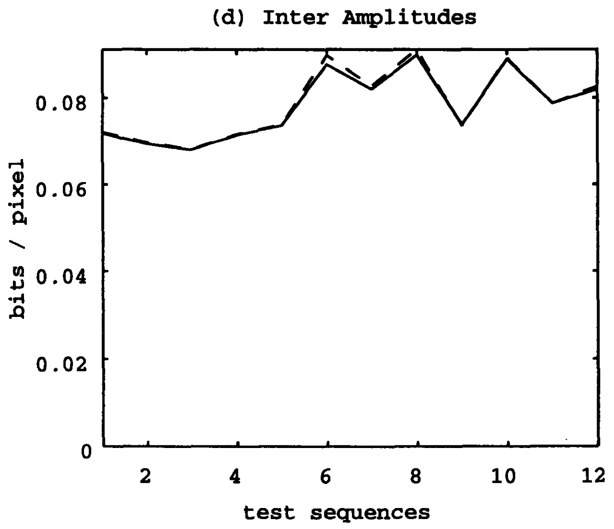


_ _ _ single-codebook separate encoding

____ position-dependent encoding

Figure 5-4: The PDE with Escape Codes limiting the codeword length vs. single-codebook separate and joint encoding: (a) Intra Runlengths, (b) Inter Runlengths.



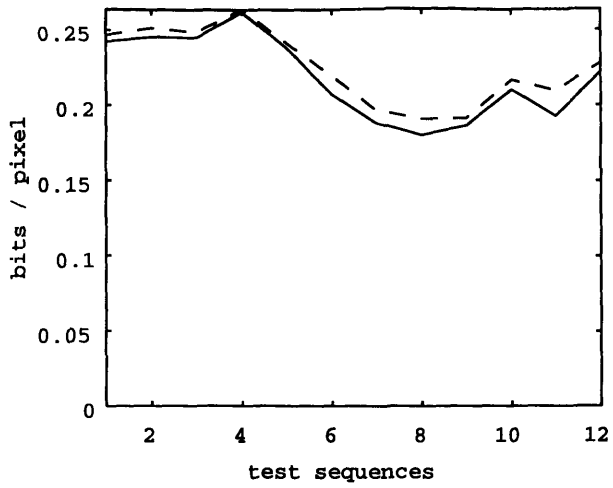


_ _ _ single-codebook separate encoding

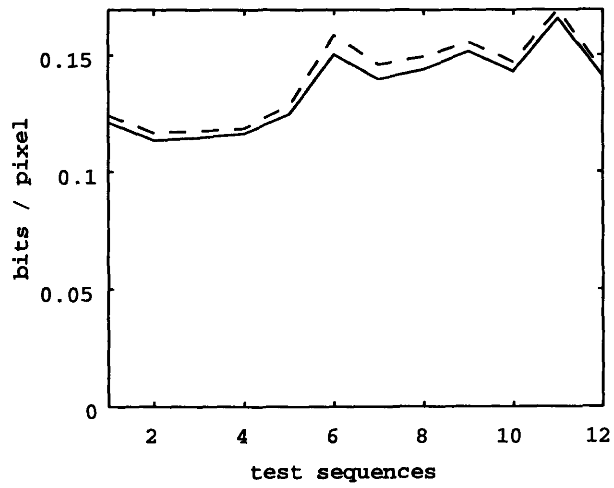
____ position-dependent encoding

Figure 5-4: (continued) The PDE with Escape Codes limiting the codeword length vs. single-codebook separate and joint encoding: (c) Intra Amplitudes, (d) Inter Amplitudes.

(e) Runlengths Total



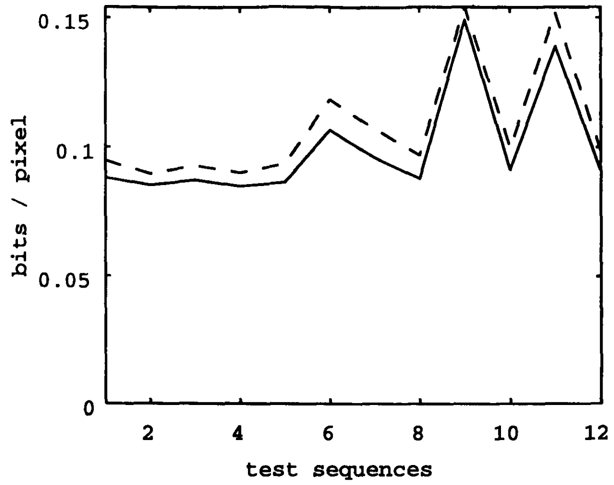
(f) Amplitudes Total



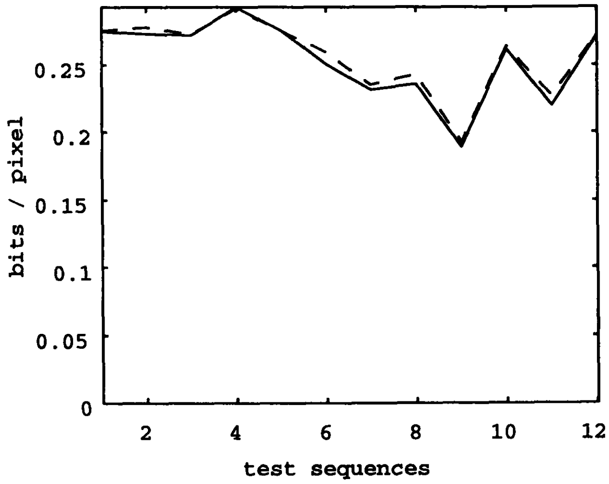
\- - - single-codebook separate encoding

——— position-dependent encoding

Figure 5-4: (continued) The PDE with Escape Codes limiting the codeword length vs. single-codebook separate and joint encoding: (e) Runlengths Total, (f) Amplitudes Total.



(g) Intra Total



(h) Inter Total

_ _ _ single-codebook separate encoding

____ position-dependent encoding

Figure 5-4: (continued) The PDE with Escape Codes limiting the codeword length vs. single-codebook separate and joint encoding: (g) Intra Total, (h) Inter Total.

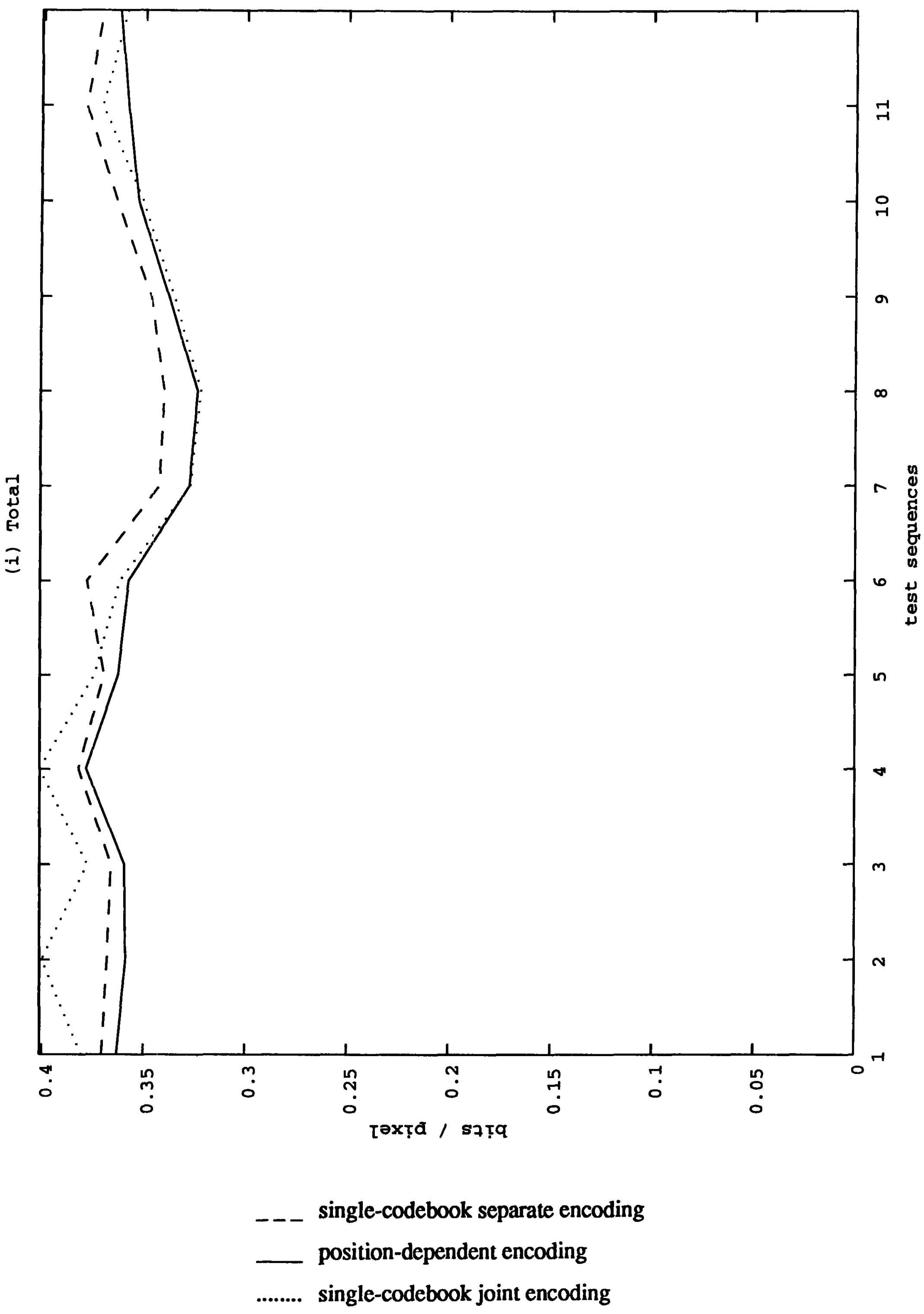


Figure 5-4: (continued) The PDE with Escape Codes limiting the codeword length vs. single-codebook separate and joint encoding: (i) Total.

5.5 The Position-Dependent Encoding with Escape Codes Limiting the Codebook Size

It was pointed out in Section 5-3, as well as Section 4-2, that memory is an important constraint. To further decrease the memory needed and to simplify the implementation, the number of entries in each codebook was limited to a preselected number.

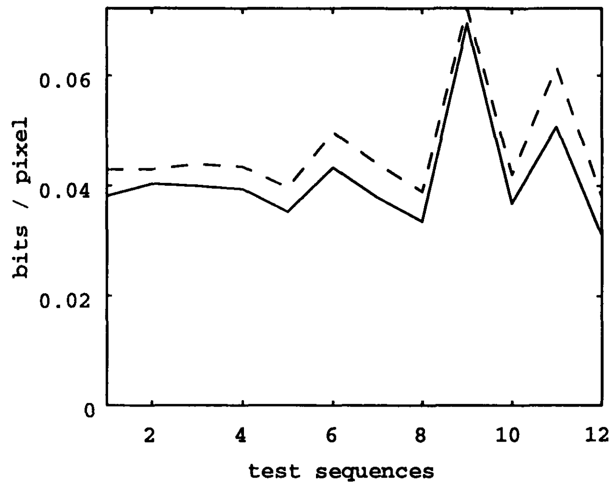
In the experiments presented here the number of entries in both runlength and amplitude codebooks was set at fifteen, with an additional entry for the escape codes. The first fifteen entries of the complete codebook (i.e. one that was not limited by the number of entries) were kept in the new codebook, while all others were escape coded. A sample codebook is shown in Appendix B.

The performance results for the 108-codebook PDE with escape codes chosen in the above manner are summarized in Table 5-5 and Figures 5-5 (a)-(i). Of course, the PDE with escape codes chosen to limit the codebook size performs worse than the PDE without escape codes (by a difference of 1.8% in the performance gain). However, it still performs better than either of the two single-codebook approaches both on average (4.3% improvement over the single-codebook separate encoding, and 3.9% improvement over the single-codebook joint encoding) and sequence by sequence. The approach taken here performs much better than the escape coding approach taken in Section 5-4, despite its simplicity. If the specific coding application requires that escape codes are introduced into the PDE scheme, then, this is the approach to be taken.

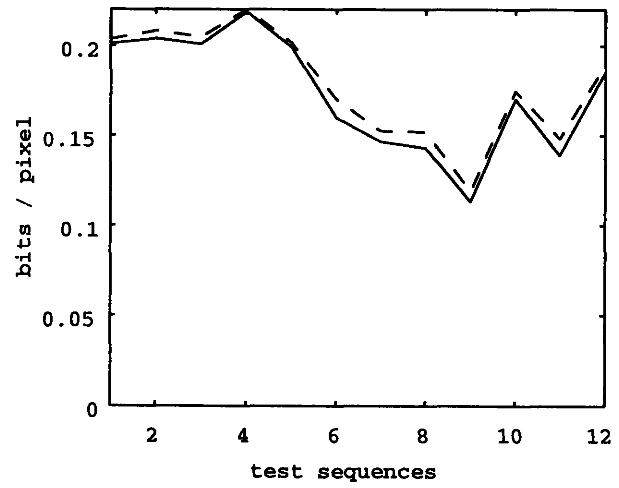
Table 5-5: The Position-Dependent Encoding with Escape Codes obtained by limiting the codebook size vs. single-codebook separate and joint encoding. The PDE uses 94 runlength and 14 amplitude codebooks.

		source	60 Hz Video						24 Hz Film				Synthetic		
		sequence	1	2	3	4	5	6	7	8	9	10	11	12	average
Runlengths	Intra	separate (bits/pixel)	.0429	.0430	.0439	.0434	.0397	.0497	.0440	.0390	.0723	.0421	.0616	.0382	.0467
		PDE (bits/pixel)	.0381	.0403	.0399	.0394	.0352	.0433	.0380	.0337	.0695	.0368	.0507	.0312	.0414
		improvement (%)	**12.7**	**6.6**	**9.9**	**10.3**	**12.5**	**14.9**	**15.9**	**16.0**	**4.1**	**14.2**	**21.3**	**22.3**	**12.8**
	Inter	separate (bits/pixel)	.2035	.2083	.2047	.2201	.2016	.1698	.1524	.1519	.1189	.1747	.1482	.1894	.1786
		PDE (bits/pixel)	.2010	.2039	.2007	.2187	.1991	.1597	.1466	.1430	.1130	.1702	.1389	.1852	.1733
		improvement (%)	**1.3**	**2.1**	**2.0**	**0.6**	**1.2**	**6.3**	**4.0**	**6.2**	**5.3**	**2.6**	**6.7**	**2.3**	**3.1**
	Total	separate (bits/pixel)	.2465	.2513	.2486	.2636	.2413	.2195	.1964	.1909	.1912	.2167	.2098	.2276	.2253
		PDE (bits/pixel)	.2391	.2442	.2406	.2581	.2344	.2030	.1845	.1767	.1824	.2070	.1897	.2164	.2147
		improvement (%)	**3.1**	**2.9**	**3.3**	**2.1**	**2.9**	**8.1**	**6.5**	**8.1**	**4.8**	**4.7**	**10.6**	**5.2**	**4.9**
Amplitudes	Intra	separate (bits/pixel)	.0519	.0469	.0491	.0469	.0542	.0687	.0633	.0582	.0818	.0579	.0906	.0607	.0608
		PDE (bits/pixel)	.0492	.0440	.0464	.0447	.0507	.0627	.0575	.0539	.0777	.0540	.0866	.0589	.0572
		improvement (%)	**5.4**	**6.5**	**5.8**	**4.7**	**6.9**	**9.7**	**10.0**	**8.0**	**5.3**	**7.2**	**4.6**	**3.0**	**6.4**
	Inter	separate (bits/pixel)	.0721	.0697	.0682	.0716	.0739	.0896	.0827	.0910	.0737	.0892	.0788	.0825	.0786
		PDE (bits/pixel)	.0718	.0693	.0681	.0714	.0736	.0875	.0815	.0893	.0733	.0884	.0776	.0817	.0778
		improvement (%)	**0.4**	**0.6**	**0.2**	**0.2**	**0.5**	**2.4**	**1.4**	**1.9**	**0.5**	**0.9**	**1.5**	**1.1**	**1.0**
	Total	separate (bits/pixel)	.1240	.1166	.1173	.1184	.1282	.1584	.1460	.1492	.1555	.1471	.1694	.1432	.1394
		PDE (bits/pixel)	.1210	.1133	.1144	.1162	.1243	.1502	.1390	.1432	.1511	.1424	.1642	.1406	.1350
		improvement (%)	**2.5**	**2.8**	**2.5**	**2.0**	**3.1**	**5.5**	**5.0**	**4.2**	**3.0**	**3.2**	**3.2**	**1.9**	**3.3**
Total	Intra	separate (bits/pixel)	.0949	.0898	.0930	.0903	.0939	.1185	.1073	.0972	.1542	.0999	.1522	.0989	.1075
		PDE (bits/pixel)	.0874	.0843	.0863	.0841	.0860	.1060	.0955	.0876	.1472	.0908	.1373	.0901	.0986
		improvement (%)	**8.6**	**6.5**	**7.7**	**7.3**	**9.2**	**11.8**	**12.4**	**11.1**	**4.7**	**10.0**	**10.8**	**9.7**	**9.1**
	Inter	separate (bits/pixel)	.2756	.2780	.2729	.2917	.2755	.2594	.2351	.2429	.1926	.2639	.2270	.2719	.2572
		PDE (bits/pixel)	.2728	.2732	.2688	.2901	.2727	.2472	.2281	.2323	.1863	.2586	.2165	.2669	.2511
		improvement (%)	**1.0**	**1.8**	**1.5**	**0.5**	**1.0**	**4.9**	**3.1**	**4.5**	**3.4**	**2.0**	**4.8**	**1.9**	**2.4**
	Total	separate (bits/pixel)	.3705	.3679	.3659	.3820	.3694	.3779	.3424	.3401	.3468	.3638	.3791	.3708	.3647
		PDE (bits/pixel)	.3601	.3576	.3551	.3742	.3587	.3532	.3236	.3199	.3335	.3495	.3538	.3570	.3497
		improvement (%)	**2.9**	**2.9**	**3.0**	**2.1**	**3.0**	**7.0**	**5.8**	**6.3**	**4.0**	**4.1**	**7.2**	**3.9**	**4.3**
		joint (bits/pixel)	.3806	.4001	.3775	.4012	.3742	.3618	.3273	.3223	.3355	.3509	.3713	.3591	.3635
		PDE vs. joint (%)	**5.7**	**11.9**	**6.3**	**7.2**	**4.3**	**2.4**	**1.2**	**0.7**	**0.6**	**0.4**	**4.9**	**0.6**	**3.9**

(a) Intra Runlengths



(b) Inter Runlengths

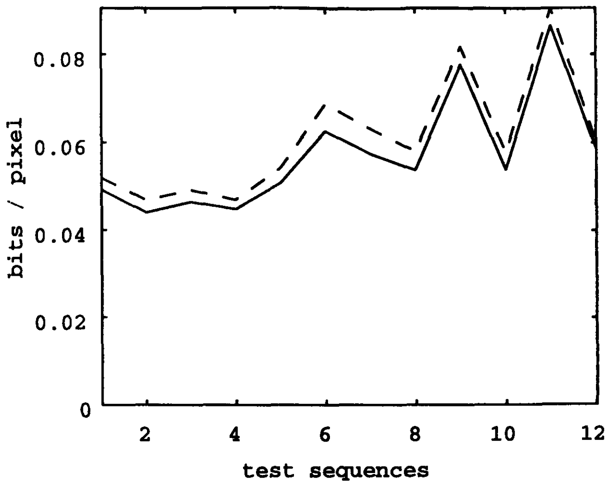


_ _ _ single-codebook separate encoding

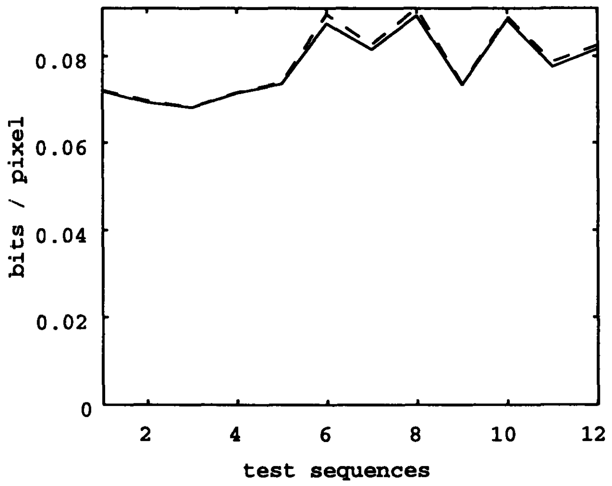
____ position-dependent encoding

Figure 5-5: The PDE with Escape Codes limiting the codebook size vs. single-codebook separate and joint encoding: (a) Intra Runlengths, (b) Inter Runlengths.

(c) Intra Amplitudes



(d) Inter Amplitudes

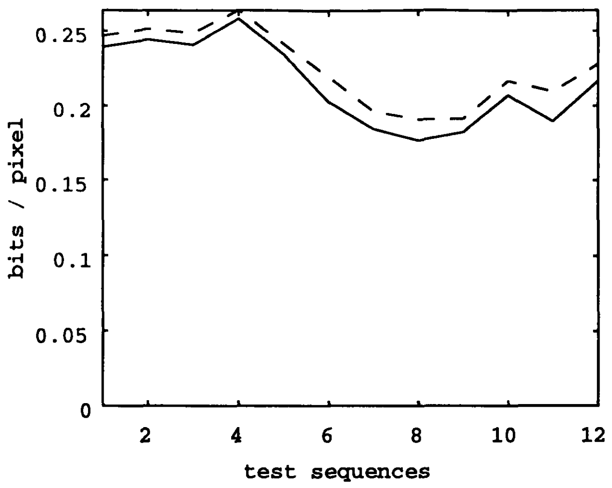


_ _ _ single-codebook separate encoding

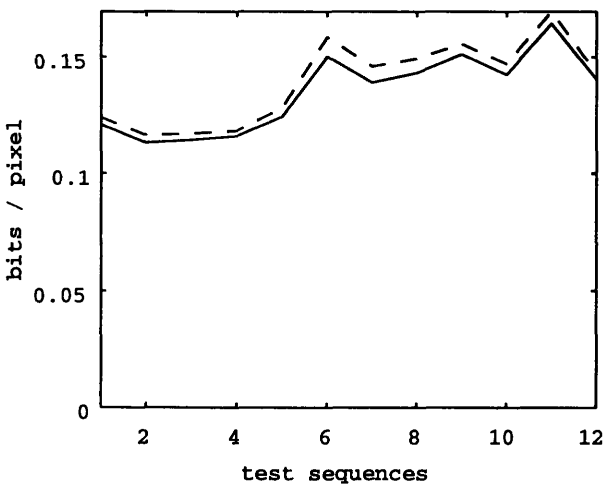
____ position-dependent encoding

Figure 5-5: (continued) The PDE with Escape Codes limiting the codebook size vs. single-codebook separate and joint encoding: (c) Intra Amplitudes, (d) Inter Amplitudes.

(e) Runlengths Total



(f) Amplitudes Total

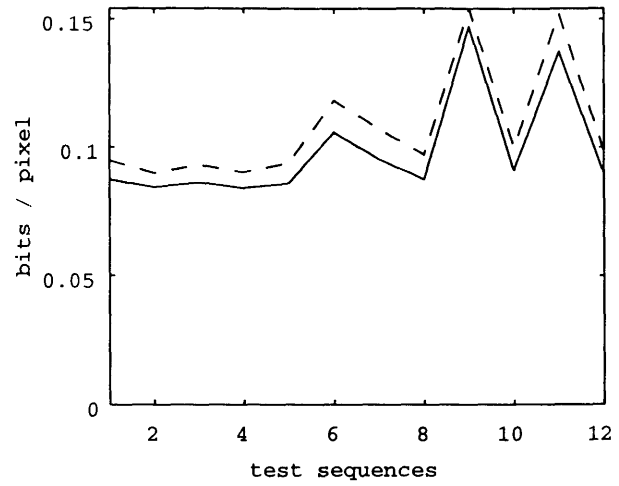


_ _ _ single-codebook separate encoding

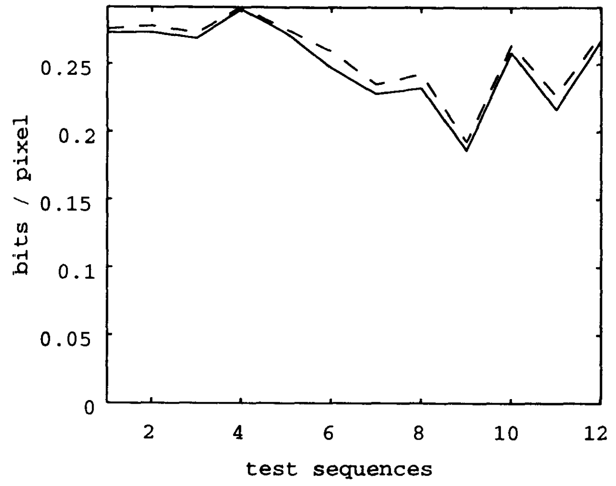
____ position-dependent encoding

Figure 5-5: (continued) The PDE with Escape Codes limiting the codebook size vs. single-codebook separate and joint encoding: (e) Runlengths Total, (f) Amplitudes Total.

(g) Intra Total



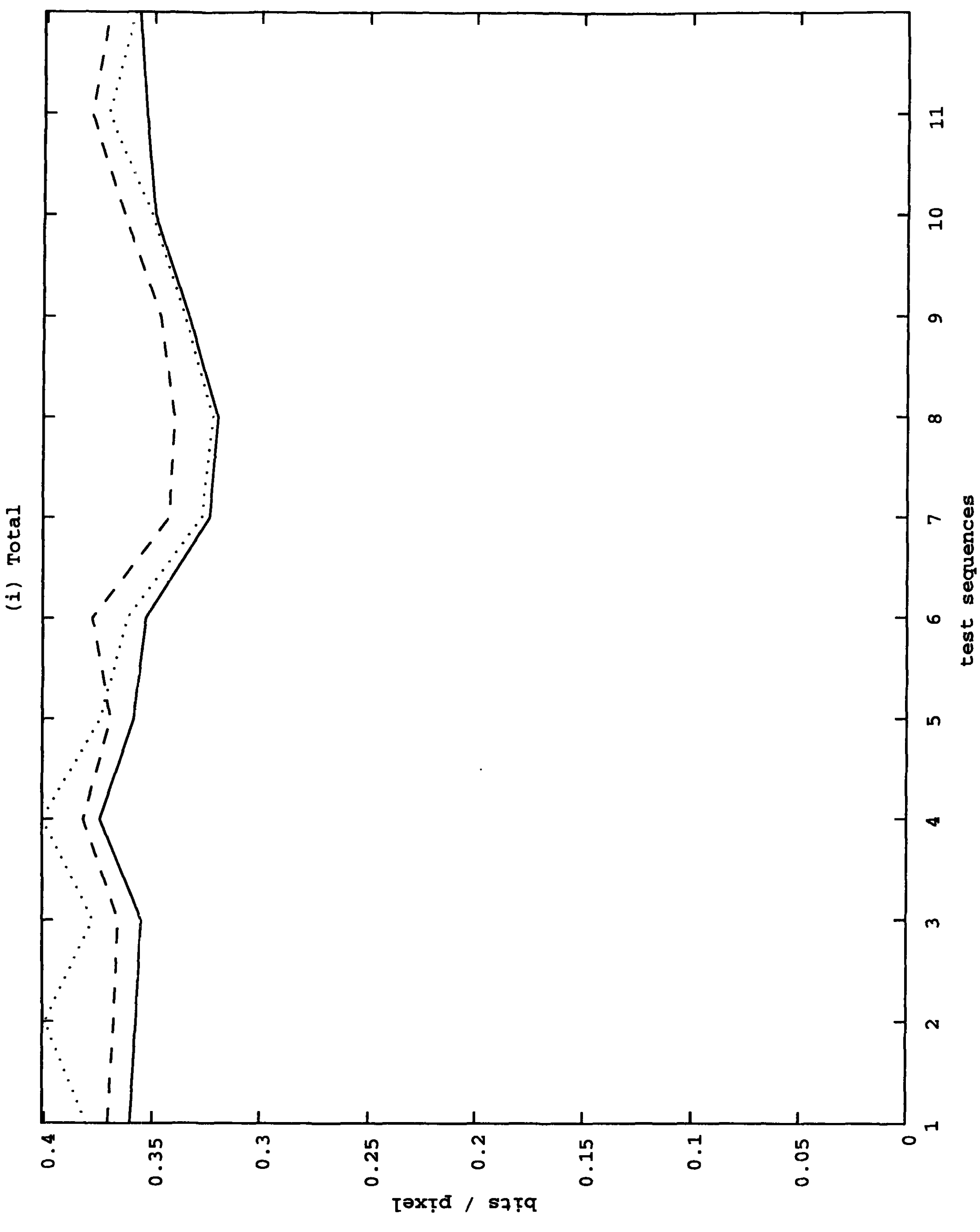
(h) Inter Total



_ _ _ single-codebook separate encoding

____ position-dependent encoding

Figure 5-5: (continued) The PDE with Escape Codes limiting the codebook size vs. single-codebook separate and joint encoding: (g) Intra Total, (h) Inter Total.



– – – single-codebook separate encoding

—— position-dependent encoding

.... single-codebook joint encoding

Figure 5-5: (continued) The PDE with Escape Codes limiting the codebook size vs. single-codebook separate and joint encoding: (i) Total.

Chapter 6

Conclusions and Future Research

Comments and Reservations

A couple of comments in regard to the tests performed are in order. Most importantly, the appropriate performance test should be (1) to train the codebooks with one set of sequences and (2) to examine the performance on another set of different sequences. This is the desired test, since it best emulates the training and running of an eventual encoder. However, the results presented here, as was already pointed out in Section 5-1, were obtained by testing the encoding scheme on the same set of sequences that was used for training of the codebooks. During the testing, the goal was to produce the best set of codebooks given the sequences that were available. Therefore, all the video sequences available were used for the training and then for the testing. The end result is that the PDE **bit rates** listed may be optimistic as compared to what would be achieved in an actual coding environment.

The PDE and the single-codebook separate encoding were both trained and tested in the exact same manner (i.e. with the same 12 sequences), and therefore the **comparison** between the two is entirely valid.

As far as the comparison between the PDE and the single-codebook joint encoding is concerned, two reservations are in order. First, while the PDE codebooks were optimized for the test sequences, the joint encoding codebooks were not (since they were prescribed by the MPEG2 standard [5]). Second, the MPEG2 codebooks were designed with the use of the B frames as well as the I and P frames. As was already pointed out in Section 5-1, no B frames were used in tests performed here. Of course, the PDE approach could be extended to include additional codebooks specifically designed for B frames.

To conclude, while the absolute bit rate numbers for the PDE may be optimistic, the comparison between the PDE and the single-codebook separate encoding is entirely valid, and the comparison between the PDE and the single-codebook joint encoding should be observed while keeping the two reservations mentioned above in mind.

The Importance of the Results

The results obtained are of great significance. The average reduction in the bit rate obtained by using the proposed PDE is about 6% (4% for the PDE with escape codes). Typical HDTV bit rates are 20 Mbits/second, with the quantized DCT coefficients occupying about 18 Mbits/second. That means that the proposed PDE produces average savings of over 1 Mbit/second (over 0.7 Mbits/second for the PDE with escape codes).

These are substantial savings, especially in an application where there is a hard limit on the amount of the medium (bandwidth) available. In HDTV it is of utmost importance to compress the data as much as possible while preserving an acceptable video quality. Therefore, a lossless scheme that reduces the bit rate, such as the PDE, is a great asset. The extra bits available with the PDE (1 Mbit/second) can be used for improving the video quality, or for additional services (such as additional audio channels, or data services). As a matter of fact, 1 Mbit/second is an equivalent of 8 independently-coded CD-quality audio channels [4].

Broad Applicability of the PDE

As was already suggested in Chapter 1, the results obtained using the PDE in a video compression context are immediately extendible to an **image compression** context. Namely, the PDE already provides for coding of the intra-mode blocks, which is the only type of blocks present in image compression. Moreover, the percentage decrease in the bit rate should be even larger for image coding than for video coding. It was discussed in Section 5.2, and can be seen from the results reported in Sections 5.3, 5.4, and 5.5 as well, that intra-mode blocks yield consistently larger performance improvement than do inter-mode blocks. The expected decrease in the bit rate in an image coding environment of the PDE over the single-codebook separate encoding should be about 10% based on the Intra block results from Section 5.3.

The position-dependent encoding presented in this thesis was developed as an extension of the separate runlength and amplitude encoding approach. The idea of introducing multiple codebooks is equally applicable to the joint runlength / amplitude encoding approach. In the **position-dependent joint encoding** approach each runlength / amplitude event would select a codebook based on the starting position of the run. Differences in statistics as a function of position of the joint runlength / amplitude events are a simple extension of the differences in statistics of the separate runlength and amplitude events discussed in Chapters 2 and 3. In the low-frequency region of a quantized DCT coefficient block, the most likely event is a short runlength followed by a large amplitude. In the high-frequency region the most likely event is a long runlength followed by a small amplitude. Given that the joint encoding codebooks are two-dimensional and, therefore, large and complex, the possibility of having many such codebooks may seem overwhelming. However, introduction of escape codes would solve the problems by limiting the codebooks only to a small number of entries.

The compression scheme, within which the PDE approach was tested, was based on the Block DCT. The idea of using multiple codebooks, based on the position of the coefficient to be encoded on a frequency-domain grid, can be applied to any compression scheme using a spatial-domain to frequency-domain transformation (e.g. wavelet transform, subband transform). Moreover, it is adaptable to other compression applications, such as audio, or speech.

Future Research

The immediate goal of future research should be to perform the tests suggested at the beginning of this chapter, i.e. to test the PDE codebooks on a set of sequences different than the training set. Also, both the PDE and the joint encoding codebooks should be trained on the same set of sequences.

Open to further investigation is the important problem of fully describing the trade-off between the number of codebooks and performance. The parameters would not only be the total number and size of the codebooks, but also the distinction between intra and inter blocks, the luminance and the chrominance, progressive and interlaced scanning, different sequence sources, etc. These questions are intimately tied to the implementation issues, but possibly some general conclusions can be drawn, and guidelines established for finding an optimal trade-off. The optimality of a trade-off, however, is also ultimately defined by the constraints of a specific application.

Appendix A

Distribution of Codebooks for the Proposed Position-Dependent Encoding

This appendix contains the exact distribution of codebooks for coding of both runlengths and amplitudes for the Proposed PDE scheme, as described in Section 5.3. The codebook distribution is, of course, the same for the PDE scheme with escape codes introduced to limit the codeword length (described in Sections 4.1 and 5.4), and the PDE scheme with escape codes introduced to limit the codebook size (described in Sections 4.2 and 5.5).

k2

22	23	28	28	30	31	31	31
15	22	23	28	29	30	31	31
15	16	21	24	27	29	30	31
10	14	16	21	24	27	29	30
9	10	14	17	20	25	27	29
3	8	11	13	17	20	25	26
2	4	7	11	13	18	19	26
	1	5	6	12	12	18	19

k1

(a) Intra Y Runlengths

Figure A-1: Distribution of runlength codebooks for the Proposed PDE scheme for: (a) Intra Y runlengths.

k2

43	43	45	45	46	46	46	46
40	43	43	45	45	46	46	46
40	40	42	43	45	45	46	46
37	39	40	42	43	44	45	46
36	37	39	41	42	44	44	45
34	36	37	39	41	42	44	44
33	34	36	38	39	41	42	44
	32	35	35	38	38	41	42

k1

(b) Intra UV Runlengths

k2

69	70	75	75	77	78	78	78
62	69	70	75	76	77	78	78
62	63	68	71	74	76	77	78
57	61	63	68	71	74	76	77
56	57	61	64	67	72	74	76
50	55	58	60	64	67	72	73
49	51	54	58	60	65	66	73
47	48	52	53	59	59	65	66

k1

(c) Inter Y Runlengths

k2

91	91	93	93	94	94	94	94
88	91	91	93	93	94	94	94
88	88	90	91	93	93	94	94
85	87	88	90	91	92	93	94
84	85	87	89	90	92	92	93
82	84	85	87	89	90	92	92
81	82	84	86	87	89	90	92
79	80	83	83	86	86	89	90

k1

(d) Inter UV Runlengths

Figure A-1: (continued) Distribution of runlength codebooks for the Proposed PDE scheme for: (b) Intra UV runlengths, (c) Inter Y runlengths, (d) Inter UV runlengths.

k2

3	3	3	3	3	3	3	3
3	3	3	3	3	3	3	3
3	3	3	3	3	3	3	3
3	3	3	3	3	3	3	3
3	3	3	3	3	3	3	3
2	3	3	3	3	3	3	3
1	2	3	3	3	3	3	3
	1	2	3	3	3	3	3

k1

(a) Intra Y Amplitudes

k2

6	6	6	6	6	6	6	6
6	6	6	6	6	6	6	6
6	6	6	6	6	6	6	6
6	6	6	6	6	6	6	6
6	6	6	6	6	6	6	6
5	6	6	6	6	6	6	6
4	5	6	6	6	6	6	6
	4	5	6	6	6	6	6

k1

(b) Intra UV Amplitudes

k2

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
9	10	10	10	10	10	10	10
8	9	10	10	10	10	10	10
7	8	9	10	10	10	10	10

k1

(c) Inter Y Amplitudes

Figure A-2: Distribution of amplitude codebooks for the Proposed PDE scheme for: (a) Intra Y amplitudes, (b) Intra UV amplitudes, (c) Inter Y amplitudes.

k2

14	14	14	14	14	14	14	14
14	14	14	14	14	14	14	14
14	14	14	14	14	14	14	14
14	14	14	14	14	14	14	14
14	14	14	14	14	14	14	14
13	14	14	14	14	14	14	14
12	13	14	14	14	14	14	14
11	12	13	14	14	14	14	14

k1

(d) Inter UV Amplitudes

Figure A-2: (continued) Distribution of amplitude codebooks for the Proposed PDE scheme for: (d) Inter UV amplitudes.

Appendix B

Sample Codebooks

This appendix contains a sample codebook for the Proposed PDE scheme, as well as for the two PDE schemes with escape codes. The sample codebook included for the three schemes is runlength codebook 16 (see Figure A-1). Note that because of the position of this particular codebook, runlengths of 40 through 63 are not possible. Therefore, the only events that have codewords are EOB and runlengths 0 through 39.

As discussed in Section 4.1, in the final codebook for the PDE with escape codes limiting the codeword length (see Table B-2) there are no entries with codewords longer than the uniform-coder codeword length (in this case 6). In the codebook for the PDE with escape codes limiting the codebook size (see Table B-3) there are a total of 16 entries (15 runlengths and the escape code). In both cases the bit sequences for the escape-coded events consist of the escape code followed by the appropriate uniform-coder codewords.

Table B-1: Runlength codebook 16 for the Proposed PDE. No escape codes used.

Runlength	# of bits	codeword
EOB	4	1111
0	1	0
1	3	110
2	4	1011
3	4	1001
4	5	11100
5	5	10000
6	7	1110110
7	7	1010110
8	6	100010
9	7	1110111
10	6	100011
11	6	101010
12	6	101000
13	6	111010
14	8	10101110
15	8	10100101
16	9	101001111
17	9	101001001
18	9	101001110
19	9	101001101
20	10	1010111101
21	10	1010010000
22	10	1010111110
23	10	1010011000
24	10	1010111100
25	10	1010010001
26	12	101011111110
27	13	1010111111110
28	12	101011111101
29	12	101001100111
30	11	10100110010
31	12	101011111100
32	13	1010111111111
33	19	1010011001100100001
34	13	1010011001101
35	19	1010011001100100000
36	15	101001100110011
37	14	10100110011000
38	19	1010011001100100011
39	19	1010011001100100010
40 - 63	not possible	

Table B-2: (a) Runlength codebook 16 for the PDE with Escape Codes introduced to limit the codeword length. (b) Complete list of all possible runlength events with their respective bit sequences.

(a)

Runlength	# of bits	codeword
EOB	4	1111
0	1	0
1	3	110
2	4	1011
3	4	1000
4	5	11100
5	6	111011
8	6	100100
10	6	100101
11	6	100111
12	6	100110
13	6	111010
Escape code	4	1010

(b)

Runlength	# of bits	codeword
EOB	4	1111
0	1	0
1	3	110
2	4	1011
3	4	1000
4	5	11100
5	6	111011
6	10	1010 + 000110
7	10	1010 + 000111
8	6	100100
9	10	1010 + 001001
10	6	100101
11	6	100111
12	6	100110
13	6	111010
14	10	1010 + 001110
15	10	1010 + 001111
16	10	1010 + 010000
17	10	1010 + 010001
18	10	1010 + 010010
19	10	1010 + 010011
20	10	1010 + 010100
21	10	1010 + 010101
22	10	1010 + 010110
23	10	1010 + 010111
24	10	1010 + 011000
25	10	1010 + 011001
26	10	1010 + 011010
27	10	1010 + 011011
28	10	1010 + 011100
29	10	1010 + 011101
30	10	1010 + 011110
31	10	1010 + 011111
32	10	1010 + 100000
33	10	1010 + 100001
34	10	1010 + 100010
35	10	1010 + 100011
36	10	1010 + 100100
37	10	1010 + 100101
38	10	1010 + 100110
39	10	1010 + 100111
40 - 63	not possible	

Table B-3: (a) Runlength codebook 16 for the PDE with Escape Codes introduced to limit the codebook size. (b) Complete list of all possible runlength events with their respective bit sequences.

(a)

Runlength	# of bits	codeword
EOB	4	1111
0	1	0
1	3	110
2	4	1011
3	4	1001
4	5	10101
5	5	10000
6	7	1110101
7	7	1110100
8	6	100011
9	6	100010
10	6	101000
11	6	111000
12	6	101001
13	6	111001
Escape code	6	111011

(b)

Runlength	# of bits	codeword
EOB	4	1111
0	1	0
1	3	110
2	4	1011
3	4	1001
4	5	10101
5	5	10000
6	7	1110101
7	7	1110100
8	6	100011
9	6	100010
10	6	101000
11	6	111000
12	6	101001
13	6	111001
14	12	111011 + 001110
15	12	111011 + 001111
16	12	111011 + 010000
17	12	111011 + 010001
18	12	111011 + 010010
19	12	111011 + 010011
20	12	111011 + 010100
21	12	111011 + 010101
22	12	111011 + 010110
23	12	111011 + 010111
24	12	111011 + 011000
25	12	111011 + 011001
26	12	111011 + 011010
27	12	111011 + 011011
28	12	111011 + 011100
29	12	111011 + 011101
30	12	111011 + 011110
31	12	111011 + 011111
32	12	111011 + 100000
33	12	111011 + 100001
34	12	111011 + 100010
35	12	111011 + 100011
36	12	111011 + 100100
37	12	111011 + 100101
38	12	111011 + 100110
39	12	111011 + 100111
40 - 63	not possible	

References

[1] J. S. Lim, *Two-Dimensional Signal and Image Processing*. Englewood Cliffs, New Jersey: Prentice Hall, 1990.

[2] A. Netravali and B. Haskell, *Digital Pictures, Representation and Compression*. New York: Plenum Press, 1988.

[3] J. G. Apostolopoulos and J. S. Lim, "Video Compression for Digital ATV Systems," in *Motion Analysis and Image Sequence Processing*, edited by M. I. Sezan and R. L. Lagendijk. Norwell, Massachusetts: Kluwer Academic Publishers, 1993.

[4] American Television Alliance, Massachusetts Institute of Technology and General Instrument Corporation, *Channel-Compatible DigiCipher HDTV System*, April 1992.

[5] ITU-TS Rec. H.26x, *Generic Coding of Moving Pictures and Associated Audio*, 3rd working draft, July 1993.

[6] D. J. DeGall, "MPEG: A Video Compression Standard for Multimedia Applications," *Communications of the ACM*, Vol. 34, pp. 46-58, April 1991.

[7] J. G. Apostolopoulos, A. Pfajfer, H. M. Jung, and J. S. Lim, "Position-Dependent Encoding," *ICASSP Proc.*, April 1994.

[8] J. G. Apostolopoulos and J. S. Lim, *Position-Dependent Runlength Encoding*, Submission to Moving Pictures Expert Group, November 1992.