INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

Université d'Ottawa · University of Ottawa

Image and Video Scalability in the Compressed Domain

Qingwen Hu

A thesis

submitted to the School of Graduate Studies and Research

in Partial Fulfillment of the Requirements

for the Degree of

Master of Applied Science

in

Electrical Engineering

Ottawa-Carleton Institute of Electrical Engineering

Department of Electrical Engineering

Faculty of Engineering

University of Ottawa

Ottawa, Ontario, K1N 6N5

November, 1996

To my parents,

Acknowledgements

First, I would like to express my sincere gratitude to my supervisor, Dr. S. Panchanathan for his constructive suggestions, valuable guidance and encouragement during my thesis work. Also many thanks to him for his helpful advice, comments and corrections in the progress of this thesis.

I would like to thank all the past and present members of the Visual Computing and Communications Laboratory, especially M. K. Mandal, N. Gamaz and D. Wu for their help and cooperation.

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

I would also like to thank my husband and parents. Their affection and various supports are always beside me in this whole period.

Abstract

Image and video compression is becoming increasingly popular with the advent of broadband networks, high powered workstations, and compression standards. Visual data are therefore expected to be stored in the compressed format for efficient transmission and storage. In advanced visual services, further manipulations of the compressed visual information may be required. Scaling of visual data in the compressed domain has many applications. For example, a simultaneous display of four video sources in a video conferencing system requires down scaling of each video source by half, translation on these video sources and generation of a composite video sequence. Another example is the need for spatial scalable visual coding in a multi-user environment, where the visual data is transmitted and displayed at different resolutions based on the viewer's request.

Compared to the traditional spatial domain techniques, the compressed domain techniques avoid the unnecessary decompression and re-compression procedures, thus resulting in a reduced computational complexity and storage requirements. In addition, significant processing speedup may be gained because of the lower data rate in the compressed domain.

In this thesis, we have proposed two novel techniques, namely format compatible (FC) DCT and format modified (FM) DCT to implement image/video spatial scalability by operating directly on the DCT compressed data. The FC-DCT technique can be used to manipulate the standard bit streams, which maintains the formats of the processed bit streams. We have implemented the JPEG DCT domain hierarchical mode encoder based on the FC-DCT technique. On the other hand, the FM-DCT technique is simpler and has a significantly lower computational complexity. Hence, the FC-DCT and FM-DCT techniques can be chosen based on the specific needs of the applications.

We note that the FC-DCT and FM-DCT techniques are not directly applicable for manipulations of video data due to the requirement of motion estimation/compensation procedure in addition to DCT coding. Hence, we propose a macroblock type selective (MTS) encoder to implement scalable MPEG video coding using a combined FC-DCT and modified DCT domain motion compensation technique. The proposed encoder provides an elegant way to re-encode the scaled video while maintaining a good video quality. In addition, it is compatible with the standard MPEG encoder.

Contents

List of Figures

List of Tables

List of Acronyms

ISO	International Standards Organization
ITU	International Telecommunications Union
JPEG	Joint Photographic Experts Group
MPEG	Moving Pictures Experts Group
HDTV	High Definition Television
DPCM	Differential Pulse Code Modulation
DM	Delta Modulation
DCT	Discrete Cosine Transform
IDCT	Inverse DCT
KLT	Karhunen-Loeve Transform
VQ	Vector Quantization
HVS	Huamn Visual System
VLC	Variable Length Code
RLC	Run Length Coding
CCIR	Consultative Committee on International Radio
MSE	Mean Square Error
MAE	Mean Absolute Error
FSA	Fast Search Algorithm
DAT	Digital Audio Tape
GOP	Group of Pictures
I-picture	Intra-coded picture
P-picture	Predictive-coded picture
B-picture	Bidirectionally predictive-coded picture
PIT	Progresive Image Transmission
PSNR	Peak Signal to Noise Ratio
Y	Luminance component
Cr	Red chrominance component
Cb	Blue chrominance component
FC-DCT	Format Compatible DCT
FM-DCT	Format Modified DCT
MTS	Macroblock Type Selective

Chapter 1

Introduction

1.1 Image/Video Compression

The recent advances in digital signal processing, VLSI and high speed networking techniques have made video services increasingly popular. Since visual communications require transmission and storage of enormous visual information, it is necessary to use image and video compression techniques to reduce the data rate while maintaining an acceptable reconstructed image/video quality. Recently, the International Standards Organization (ISO) and the International Telecommunications Union (ITU) have proposed standards such as JPEG, MPEG and H.261 for image and video compression, respectively [1-5]. Applications include video conferencing, HDTV, multimedia databases, interactive television, mobile multimedia communications, etc.

Image/video compression is essentially a redundancy removal process. Efficient compression can be achieved by exploiting psychovisual as well as statistical redundancies such as spatial, temporal in the visual data. In a still image or a video frame, the spatial

redundancy is typically removed by employing intraframe coding techniques such as predictive coding (DPCM), transform coding (DCT), vector quantization, etc. In a video sequence, both temporal and spatial redundancies exist. Typically, the temporal redundancy can be greatly reduced by using interframe coding techniques such as motion estimation/compensation, which is followed by spatial redundancy removal techniques. Two most popularly used motion estimation techniques in the literature are pel-recursive [24, 45, 46] and block-matching [43], respectively. In the pel-recursive algorithm, motion estimation is based on individual pixels, which is a computationally intensive procedure. On the other hand, the block-matching algorithm employs block based motion estimaiton/compensation procedure, which is computationally efficient and is hence widely used. To perform motion estimation, the current frame is divided into non-overlapping blocks, which are compared with the candidate blocks in the reference frame within a search area in order to obtain the best match with respect to a pre-specified error criterion. The relative displacement between the current block and the best match is specified by a two-dimensional motion vector, which is then used for motion compensation. For both image and video compression, the psychovisual redundancy existing in the visual data can be further exploited based on the properties of the human visual system [25].

1.2 Investigated Approaches for Manipulations of Visual Data in Compressed Domain

The advent of the JPEG, MPEG and H.261 standards has resulted in visual data in various applications being stored in the compressed format for efficient transmission and storage. In

advanced visual services, further manipulations of the compressed visual information may be required. For example, in a video conferencing system, scaling, translation (block-wise or pixel-wise), overlapping (opaque or semi-transparent), linear filtering. etc.. of visual data are required [18]. In addition, scalable encoding of visual data for a multi-user environment is extremely important in applications including browsing visual databases, querying multimedia databases and interactive multimedia communications.

We note that most compression techniques such as JPEG, MPEG, H.261 are discrete cosine transform (DCT) based. A straight forward method to manipulate the coded visual data bit stream is to: (i) fully decompress the DCT data, (ii) reconstruct the visual data back to the spatial domain, (iii) perform the manipulations on the reconstructed image, (iv) re-compress the processed data. We refer to this technique as the spatial domain technique. This has several disadvantages: (i) it entails decompression and re-compression of image/video which correspond to a large computational complexity, (ii) it requires extra storage for the decompressed data, and (iii) some precision will be lost since most encoder/decoders are lossy. It is therefore desirable to manipulate the data directly in the compressed domain to avoid the unnecessary decompression and re-compression procedures resulting in a reduced computational complexity and storage requirements. In addition, significant processing speedup may be gained because of the lower data rate in the compressed domain. We refer to this technique as the DCT domain technique.

There is an increasing interest in manipulating visual information directly in the compressed domain [15-21]. In this thesis, we propose two novel techniques namely format compatible (FC) DCT and format modified (FM) DCT to implement image/video spatial scalability by operating directly on the DCT compressed data.

The FC-DCT technique can be used to manipulate the standard bit streams, such as JPEG, MPEG, etc., and generate the compatible scaled image and video bit streams without changing the basic formats, i.e. the size of the DCT block. This technique is especially useful when re-encoding of the processed data is required. The FM-DCT technique relies on the modification of the basic DCT block size for different levels of spatial scalability, and is more suitable for fast decoding. Since there is no need for maintaining the basic block size, it has a significantly lower computational complexity than the FC-DCT technique. The FM-DCT technique can be employed in a variety of applications which require fast processing, such as indexing, retrieval and browsing through different resolutions of coded image/video. We note both techniques are important and useful. The choice of the technique depends on the requirements of the specific applications.

As mentioned above, the FC-DCT technique is used for manipulating the standard bit streams. However, it is not directly applicable for video data, since motion estimation/compensation is involved for temporal redundancy removal. For example, to scale down MPEG video in the compressed domain, motion activity will change in the processed video, and thus modifications to motion estimation/compensation are required. Hence, we propose an approach to implement encoding of the scaled MPEG video using a combined FC-DCT and modified DCT domain motion compensation technique. A key step in the encoder is macroblock based motion estimation/compensation recalculation for the scaled frames. We note a new macroblock in the scaled frame corresponds to four macroblocks in the original frame. By analyzing the motion activity, we can either derive a new macroblock motion vector from existing motion vectors of four adjacent (original) macroblocks, or disregard the motion compensation and enforce a new macroblock to be intra-coded. We call this encoder as a

macroblock type selective (MTS) DCT domain encoder, which performs all of the operations in the compressed domain. The MTS-encoder is also compatible with the standard MPEG encoder. Hence it provides an elegant way to re-encode the scaled video while maintaining a good video quality. Furthermore, the proposed technique can be extended to multiple layers for implementation of MPEG spatial scalability in the MPEG compressed domain.

1.3 Thesis Organization

The rest of this thesis is organized as follows: Chapter 2 presents a review of image/video compression techniques, followed by the basic compression techniques defined in the JPEG and MPEG standards. In chapter 3, scalable image/video compression techniques including SNR scalability, spatial scalability and temporal scalability are described, and the scalable compression schemes defined in the JPEG and MPEG standards are presented. The proposed DCT domain techniques namely FC-DCT and FM-DCT for image/video spatial scalability are detailed in chapter 4. In chapter 5, the proposed MTS-encoder for encoding scaled MPEG video in the motion-compensated DCT domain is presented. This encoder is implemented using a combined FC-DCT and modified DCT domain motion compensation technique. Finally, the conclusions and suggestions for future research work are presented in chapter 6.

1.4 Main Contributions

The main contributions of this thesis are summarized as follows:

- Investigate JPEG spatial scalable compression algorithms (hierarchical mode) for image compression.

- Propose a novel format compatible (FC) DCT technique to implement JPEG DCT domain hierarchical mode algorithm, and evaluate its performance with respect to the corresponding JPEG spatial domain hierarchical mode algorithm.
- Propose a novel format modified (FM) DCT technique for visual spatial scalable coding in the compressed domain, and compare its performance with the spatial domain technique and the FC-DCT technique.
- Investigate the MPEG compression algorithm for video compression.
- Propose a novel approach to implement scalable encoding of MPEG video in the compressed domain using a combined FC-DCT and modified DCT domain motion compensation technique.

Chapter 2

Review of Image/Video Compression

In this chapter, the concepts of entropy (lossless) coding, source (lossy) coding and hybrid coding are presented. Several lossless data compression techniques are discussed in section 2.2. In section 2.3, some intraframe coding techniques are presented, such as predictive coding, transform coding, sub-band coding, vector quantization, fractal coding, object/model based coding, etc. In section 2.4, we review the JPEG standard for still image compression, and present the baseline sequential coding scheme. Some interframe coding techniques including conditional replenishment, adaptive predictive coding with motion compensation and intraframe hybrid coding are addressed in section 2.5. Finally, we present a review of video compression standards with a particular emphasis on the MPEG standard in section 2.6.

2.1 Image/Video Data Compression

The large volumes of visual information for storage and transmission necessitate the use of compression. Efficient compression can be achieved by exploiting spatial, temporal and

psychovisual redundancies. Spatial redundancy refers to the high correlation among neighboring pixels in an image/video frame, which is typically removed by employing intraframe coding techniques such as predictive coding, transform coding, etc. Temporal redundancy corresponds to the correlation between successive frames in a video sequence, where the difference between frames are caused by object motion and/or camera motion, panning, zooming, etc. Temporal redundancy can be greatly reduced by using interframe coding techniques such as motion estimation/compensation. Psychovisual redundancy refers to the feature that some information within a image/video frame can be removed without sacrificing the subjective quality. Higher compression can be achieved by exploiting the properties of the human visual system (HVS). For example, human eyes are more sensitive to changes in brightness than to color changes, and less perception of distortion at higher spatial frequencies. Therefore, in many visual compression techniques, sub-sampling of chrominance information and weighted quantization of transform coefficients are used to reduce the data rate while matching the sensitivity of HVS.

Visual compression techniques fit into three categories: entropy (lossless), source (lossy) and hybrid coding [26]. Entropy coding is a "lossless" process. In this coding scheme, a data stream is considered as a simple digital sequence and the semantics of the data are ignored. Practical implementation of entropy coding includes run-length coding, Huffman coding, arithmetic coding, etc. Since a high degree of correlation exists between neighboring pixels in natural images, the statistical redundancy can be exploited without losing any information. In other words, the entire coding process is reversible. Lossless compression techniques are especially useful in applications which require very high fidelity reconstructed images. According to Shannon's lossless coding theorem, the average encoding bit rate of an

information source is theoretically lower bounded by the information content or entropy H. Therefore, the effectiveness of entropy coding can be evaluated by comparing the coding bit rate and source entropy. For a source with L possible independent symbols with probabilities p_i $(i = 0,1,\cdots,L-1)$, its entropy is defined by:

$$H = -\sum_{i=0}^{L-1} p_i \log_2 p_i \tag{2.1}$$

Entropy coding results in variable length code (VLC), where highly probable source symbols are assigned to shorter code bits, and vice versa.

Source coding exploits the semantics of the data, where the compression efficiency is content dependent. It is a lossy process, and results in higher compression ratios compared to entropy coding. Examples include predictive coding, transform coding, sub-band coding and vector quantization. The objective of lossy coding is to minimize the average distortion while achieving high compression.

Hybrid coding is a combination of well known algorithms and transform coding. For example, transform coding can be combined with entropy coding, predictive coding, etc. Most image and video compression standards like JPEG, MPEG, H.261 employ hybrid coding schemes.

2.2 Lossless Data Compression Techniques

We note several visual compression standards such as JPEG, MPEG. H.261, etc., use a transform codec combined with lossless coding schemes to achieve higher compression ratios. For example, the JPEG standard uses run-length coding technique to encode block DCT coefficients. and the coded coefficients are further compressed using either arithmetic coding or

Huffman coding. Similarly, both the MPEG and H.261 standards employs run-length coding and Huffman coding on the block DCT coefficients. In this section, several lossless data compression techniques are discussed.

2.2.1 Run-length Coding

In run-length coding (RLC), a sequence of consecutive pixels are encoded by sequentially transmitting the pixel values followed by the number (run) of the identical values. RLC is especially suitable for binary image compression such as text, graphics, weather maps, etc., where the long runs of white pixels in the images result in significant bit rate reduction. RLC is not efficient for compressing images with high detail. However, RLC can be used to provide high compression when combined with transform technique. In a transform codec, after quantization of transform coefficients, many coefficients become zeros. Therefore, long runs of zero valued coefficients are expected, and two-dimensional RLC followed by Huffman or arithmetic coding is applied to efficiently compress the coefficients.

2.2.2 Huffman Coding

The most popular entropy coding techniques are Huffman coding and arithmetic coding. Huffman coding is easier to implement compared to arithmetic coding. The Huffman code for a source is formed by multiple stages. The coding procedure is summarized as follows: (i) arrange the source symbols in the decreasing order of their probabilities with each symbol considered as a node, (ii) in each stage, combine the two least probable symbols (nodes) into a new node whose probability is the sum of the merged symbols, (iii) assign "0" and "1" respectively to these two symbols, (iv) repeat this process until the source with only one node is

in the last stage [30]. The Huffman code for each symbol of a source is therefore generated by passing sequentially from the root node to the leaf node (symbol). An example of Huffman coding is illustrated in Fig. 2.1.

The design of Huffman code is source statistical dependent. Typically, Huffman coding would pre-design a different codeword set for different input statistics. Since the same codebook is stored in both the encoder and decoder, Huffman encoding and decoing can be simply implemented as a table look-up procedure.

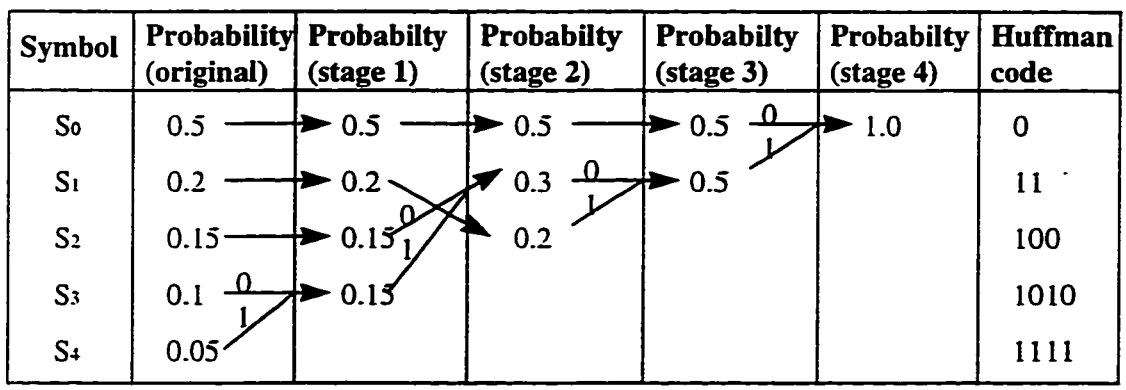


Fig. 2.1. Huffman coding procedure

2.2.3 Arithmetic Coding

Arithmetic coding achieves a higher compression performance than Huffman coding. However, arithmetic coding differs considerably from the more familiar Huffman coding, and has a higher computational complexity.

Arithmetic coding encodes the data string by creating a code string which represents a fractional value on the number line between 0 and 1 [27-29]. The coding algorithm is recursive, i.e. it operates upon one data symbol per recursion. On each recursion, the algorithm

successively partitions an interval of the number line between 0 and 1, and retains one of the partitions as the new interval. Thus, the algorithm successively deals with a smaller line. The code string, viewed as a magnitude, lies in each of the nested intervals.

We will now give an example of multi-alphabet arithmetic coding. Table 2.1 illustrates the probabilities and the cumulative probabilities of the symbols: a, b c, d.

In Fig. 2.2, we represent the codewords of Table 2.1 as points on the unit interval. The four code points divide the unit interval into four subintervals. Each codeword (code point) is the sum of the probabilities of the preceding symbols, i.e. the cumulative probability of the symbol. The width or size of the subinterval to the right of each code point corresponds to the probability of the symbol.

Table 2.1. Probabilities of the input symbols

Symbol	Probability p (in binary)	Cumulative probability P (in binary)
a	.100	.000
b	.010	.100
c	.001	.110
d	.001	.111

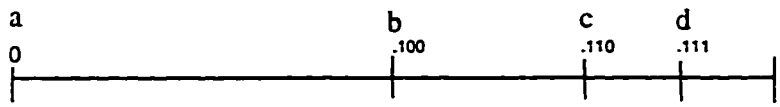


Fig. 2.2. Codewords as points on the unit interval in arithmetic coding

In Fig. 2.3, we represent the arithmetic coding process of sub-dividing the unit interval recursively. Let C and A represent the leftmost point and the interval width in the coding

process. At the beginning of encoding, the code point is at 0, and the interval is 1. When coding a symbol, the new leftmost point of the new interval is the sum of the current code point C, and the product of the current interval width A and the cumulative probability P for the symbol being coded. The new interval is the product of the probability p of the symbol being coded and the current interval width A [27].

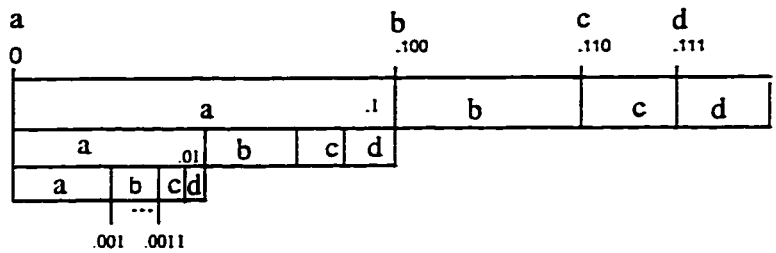


Fig. 2.3. Successive subdivision of the unit interval for data string "aab..."

$$\text{New } C = \text{Current } C + (\text{Current } A \cdot P) \tag{2.2}$$

$$\text{New } A = \text{Current } A \cdot p \tag{2.3}$$

For a data string "aab", we can use (2.2) and (2.3) recursively to form the codewords. In Fig. 2.2, after the first "a" is coded, the interval is [0, .1). After the second "a" is coded, the interval becomes [0,.01). After "b" is coded the interval becomes [.001,.0011). In fact any fraction in the range of [.001,.0011) can represent the input string "aab". It's not necessary for the decoder to know both ends of the range produced by the encoder. Instead, a single number within the range will suffice. For example, we can use .001 to represent the coded string, others such as .00101,.001011 can also do. We can decode by magnitude comparison. For this example, the code string is less than .1, so the first symbol must be "a", and so on. Since the

decoder can recursively recreate from the code string how the encoder must have successively partitioned and retained each nested subinterval, the data string can be fully recovered.

Arithmetic coding enables greater reductions in bit rate than achieved by an optimum Huffman coder, since it generates a code string which is the arithmetic combination of probabilities of individual symbols, and the average number of bits per symbol can therefore be a non-integer. However, in Huffman coding, each symbol must be coded with an integer codeword.

2.3 Intraframe Techniques

In an image or a video frame, high spatial redundancy exists among adjacent pixels. Techniques that exploit only spatial redundancy are classified as intraframe schemes. In this section, we will introduce several important intraframe coding techniques including predictive coding, transform coding, sub-band coding, vector quantization, fractal coding, object/model based coding and hybrid coding.

2.3.1 Predictive Coding

Predictive coding can efficiently exploit the mutual redundancy between neighboring pixels and encode only the new information. To differentially encode a pixel, its value is first predicted from the previous encoded pixels, and then the difference between the prediction and the actual pixel value called the prediction error is generated for encoding. For highly correlated data, the prediction errors contain little information, and hence resulting in high compression.

DPCM

Differential Pulse Code Modulation (DPCM) is a widely used technique for predictive coding. The block diagram of a DPCM system is illustrated in Fig. 2.4.

In the transmitter, there are two components, i.e. quantizer and predictor. The predictor estimates the value of current pixel using the previous encoded pixels. The predicted value $\tilde{x}_k$ is subtracted from the actual value x_k to form the prediction error d_k, which is further quantized ($\hat{d}_k$), entropy coded and transmitted. In a DPCM codec, x'_k represents the reconstructed x_k. In general, the predictor can be either linear or nonlinear. However, a nonlinear predictor is difficult to implement due to the nonlinear combination of the previous encoded pixels. Therefore, many DPCM systems use a linear predictor.

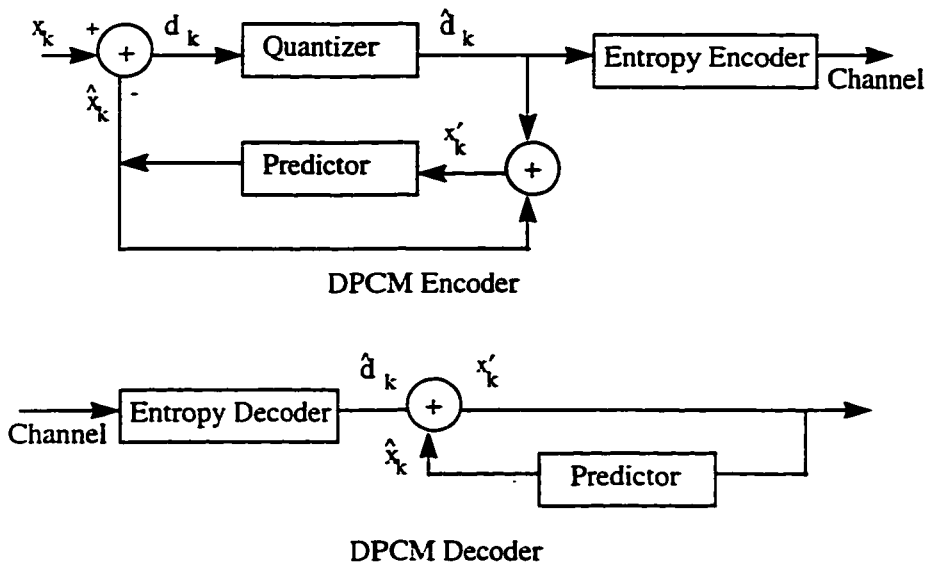


Fig. 2.4. Block diagram of a DPCM coding system

A linear prediction of x_k, i.e. $\tilde{x}_k$ is determined by equation (2.4) [31]. The optimal mean square error predictor can be obtained by choosing a_i to minimize the variance or the energy of the prediction error d_k, where the optimal coefficients a_i can be calculated using equation (2.5). For the quantizer, the quantization step size of the quantizer is designed depending on the bit rate and MSE (mean square error) requirements.

$$\tilde{x}_k = \sum_i a_i x_{k-i} \tag{2.4}$$

where a_i $(i = 0,1,\cdots)$ are weighted coefficients.

$$R(j) = \sum_i a_{i,opt} R(j-1) \tag{2.5}$$

where $R(j)$ $(j = 0,\pm1,\pm2,\cdots)$ is the autocorrelation function of the input data.

A special case of DPCM is Delta Modulation (DM), which is the simplest predictive coding. DM only encodes two level difference signals i.e. positive or negative using 1-bit quantizer.

Adaptive DPCM

DPCM technique achieves significant compression by exploiting the correlation between neighboring pixels. However, it employs a fixed predictor and quantizer, and is therefore sensitive to changes in the statistics of the data. To improve the performance of DPCM, adaptive techniques can be used to dynamically track the local statistics of the input data. An adaptive DPCM codec can be implemented by using either an adaptive predictor with a fixed quantizer or an adaptive quantizer with a fixed predictor [32].

When designing the adaptive quantizer or predictor, the respective parameters are periodically updated with changing statistics of the input data. For example, in an adaptive predictor, the autocorrelation function $R(j)$ is modified periodically based on the statistics of the data, which causes the modifications on the weighted coefficients a_i using equation (2.5).

2.3.2 Transform Coding

In transform coding, the image data is transformed from spatial domain to frequency domain via an orthogonal transform. A typical transform coding system is as shown in Fig. 2.5.

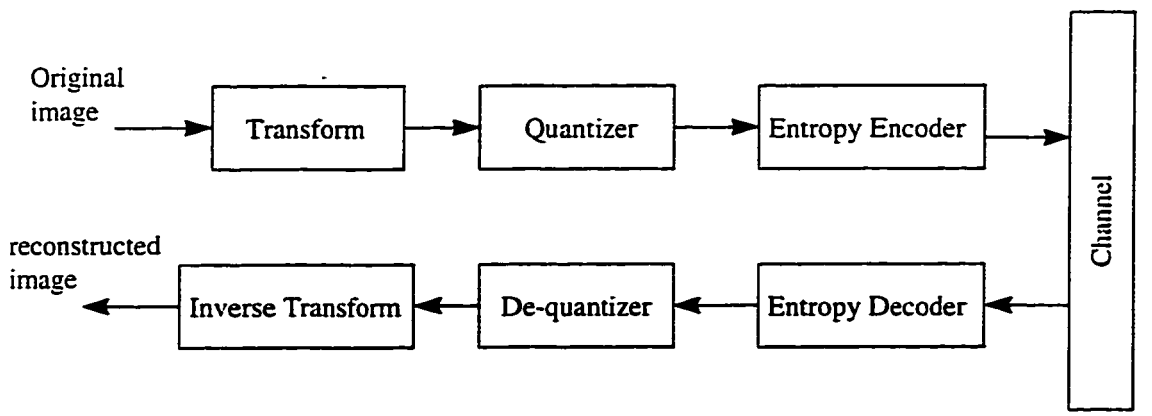


Fig. 2.5. Block diagram of a transform coding system

The transform domain data is more suitable for compression, since a large fraction of the image energy is compacted into relatively fewer low frequency transform coefficients. Typically, the low frequency coefficients contain most crucial information required for image reconstruction, hence the codec can use visually weighted quantization on the coefficients to achieve compression, where the quantized coefficients are then entropy coded to achieve further compression.

An optimal transform should result in statistically independent coefficients [42]. A well known optimal transform is Karhunen-Loeve transform (KLT) [52], which minimizes the mean square distortion of the reconstructed data for a given data rate [53]. However, KLT has a high computational complexity. Hence, it is usually replaced by sub-optimal transforms [32], such as Fourier transform, cosine transform, Hadamard transform, etc. Among them, discrete cosine transform (DCT) has been adopted in several image and video compression standards, such as JPEG, MPEG, H.261, etc., since its compression performance is closest to the optimal KLT [33].

In practice, the input image is first divided into non-overlapping blocks, and then each block is transformed independently. The choice of the block size is a tradeoff between compression efficiency and image quality [34]. Larger block sizes usually provide better compression, since more pixels are considered for redundancy reduction. However, if the block size becomes too large, the assumption of data stationarity no longer holds, therefore the degradation such as ringing and blocking artifacts are introduced. Experimental results show that block size of 8x8 or 16x16 is a good compromise.

The forward 2-D DCT of an NxN block is defined as follows [30]:

$$F(u,v) = \frac{2}{N} C(u)C(v) \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f(i,j) \cos\frac{(2i+1)u\pi}{2N} \cos\frac{(2j+1)v\pi}{2N} \tag{2.6}$$

for $u,v = 0,1,\cdots,N-1$

where $$C(x) = \begin{cases} \frac{1}{\sqrt{2}} & x = 0 \\ 1 & \text{otherwise} \end{cases}$$

$f(i,j)$ = value of pixel (i,j) in the image block

$F(u.v)$ = DCT coefficients with frequency indices $(u.v)$

In matrix notation

$$F = A^N f (B^N)^T \tag{2.7}$$

where $A^N_{u,i} = \sqrt{\frac{2}{N}} C(u) \cos\frac{(2i+1)u\pi}{2N}$, $B^N_{v,j} = \sqrt{\frac{2}{N}} C(v) \cos\frac{(2j+1)v\pi}{2N}$

$A_{u,i}$ is the element of matrix A^N at u^{th} row and i^{th} column

$B_{v,j}$ is the element of matrix B^N at v^{th} row and j^{th} column

A^N and B^N are NxN cosine transform matrices

The transform coefficient with zero frequency is called DC coefficient and remaining coefficients are called AC coefficients. The corresponding inverse 2-D DCT is defined as:

$$f(i,j) = \frac{2}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C(u)C(v) f(u,v) \cos\frac{(2i+1)u\pi}{2N} \cos\frac{(2j+1)v\pi}{2N} \tag{2.8}$$

for $i, j = 0,1,\cdots, N-1$

The cosine terms in the cosine transform matrices represent the 2-D transform kernel (basic function) of DCT. It can be seen that DCT is a linear and separable transform. For a first-order stationary Markov source model, DCT is very close to the KLT. Since natural images generally exhibit high pixel-to-pixel correlation, DCT has excellent energy compaction for visual data.

2.3.3 Sub-band Coding

Sub-band coding was first developed for speech compression. Recently, it has emerged as a powerful approach for image and video compression [35]. In sub-band coding, an image is filtered and decomposed into several sub-images or sub-bands, each of which contains a limited range of spatial frequencies. Since the sub-bands require less bandwidth than the original image, they can be down-sampled. We note the data characteristics of different bands

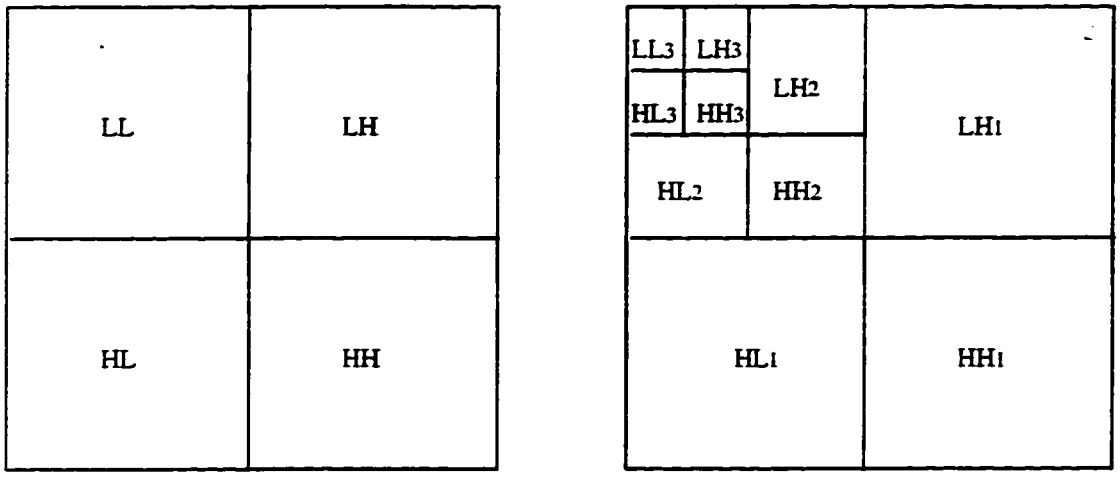
vary widely, and human visual sensitivity to degradation of different bands is not identical. hence better compression performance can be achieved by coding sub-bands differently according to their characteristics. To reconstruct the image, the decoded sub-bands are up-sampled and combined using an appropriate set of filters.

Wavelet transform is a special case of sub-band coding. It is becoming increasingly important in image compression because of its flexibility in representing nonstationary signals and its ability in adapting to the human visual system characteristics [36, 50]. This technique results in subjectively pleasing images due to the absence of blocking artifacts and reduced aliasing distortion [51]. The improved quality of reconstructed images is desirable for very low bit rate applications.

In wavelet transform, an image is decomposed into a pyramid structure of sub-images with various resolutions corresponding to different frequency bands [37, 54, 55]. Hence an image is transformed into a multi-resolution /multi-frequency representation. The sub-images localized in both space and frequency domains are relatively more stationary and thus easier to code. The representation of decompositing an image is shown in Fig. 2.6, where the two-dimensional separable wavelet transform is implemented independently, first in the horizontal direction and then in the vertical direction [37].

In a one stage wavelet transform (Fig. 2.6(a)), an image is decomposed into a low pass sub-image and three orientation selective high pass sub-images with half the original size in both the directions. The low pass sub-image (LL) is a reduced size version of the original image, while the other sub-images (LH, HL, HH) consist high frequency informaiton along the horizontal, vertical and diagonal directions, respectively. The LL sub-image can be further

decomposed by successive application of wavelet transform until the smallest size image is obtained. A three stage wavelet transform is shown in Fig. 2.6(b).



(a) One stage (b) Three stages

Fig. 2.6. Representation of two-dimensional wavelet transform

2.3.4 Vector Quantization

Vector quantization (VQ) is an efficient technique for very low bit-rate applications [38]. In VQ, the image is first partitioned into non-overlapping blocks, which is then reorganized as a vector. A fundamental result of Shannon's coding theorem shows that better compression performance can always be achieved by coding vectors instead of scalars.

A vector quantizer is defined as mapping of an L-dimensional vector V_i into another L-dimensional vector W_j (equation (2.9)), where W_j is one of a finite set called VQ codebook or VQ table of reproduction vectors (codewords). The codebook is represented by $\{W_1, W_2, \cdots, W_N\}$, where $W_i = \{w_{i1}, w_{i2}, \cdots, w_{iL}\}$, and N is the number of vectors in codebook. Therefore, vector quantization scheme partitions the L-dimensional Euclidean space into N decision regions, each containing one of the N reproduction vectors.

$$q: V_i \rightarrow W_j \tag{2.9}$$

An identical codebook exists at both the encoder and the decoder. At the encoder, each data vector is matched with the codeword in the codebook according to a minimum distortion or nearest neighbor rule. The address or index of that codeword is transmitted instead of the data vector itself resulting in high compression. At the decoder, the index is mapped back to the codeword, which is used to represent the original data vector. Hence, the reconstruction of image is simply a table look-up process.

An important step in VQ is the codebook design, which is based on a set of training vectors. A popular design approach referred to as the LBG algorithm is the generalized Lloyd clustering algorithm [39]. In this algorithm, an initial codebook is first set and each training vector is assigned to its nearest neighbor codeword. In the training stage, each codeword is updated to minimize its distortion relative to the vectors assigned to it. This process is then be executed recursively until the change in distortion between two successive iterations is within a acceptable threshold.

VQ is a powerful technique for image compression. However, there are several drawbacks. It can be seen that VQ is highly image dependent, therefore it is difficult to design a good codebook for all the possible occurrences of pixel combinations in a image block. In addition, its computational complexity grows exponentially with vector dimension. A well known degradation associated with VQ are edge jaggedness and blocking effects, especially when the codebook size is small.

2.3.5 Fractal Image Compression

Fractal image compression [75, 77] is a promising new technology. Fractal geometry resembles the geometry of images at different scales and angles. This insight opens vast possibilities for generating natural looking images.

Fractal image compression is the inverse process of fractal image generation. It uses an iterated function system (IFS) for representing images. An IFS is a set of contractive transformations that map from a defined rectangle of the image to smaller portions of that rectangle. In practice, affine transformations are used to translate, scale, shear and rotate points in the image [78, 79]. A modified scheme for representing images is called partitioned iterated function system (PIFS), which is employed to capture the diversity of real images [80, 81]. In a PIFS, the transformations do not map from the whole images to the parts, but from larger parts to smaller parts, where the large areas are called domain blocks, and the small areas are called range blocks. The essence of the compression process is the pairing of each range block to a domain block minimal based on an affine transformation, such that the difference between them is. Since a very small amount of data are required to describe each transformation, fractal coding results in very high compression [76].

Fractal compression is a lossy method, and is highly asymmetric. The compression process is computationally intensive, on the other hand, the decompression process is fast.

2.3.6 Object/Model based Coding

MPEG-4 [82, 83] is intended to provide generic method for very high compression ratios. The well-known DCT block-based techniques are robust and very well optimized. However, they are not efficient at very low bit rates, since the DCT coefficients cannot be properly coded

resulting in visible block structures. For complex sequences, these blocking effects can become very apparent.

Several algorithms such as object-based coding, model-based analysis/synthesis methods, etc., have been proposed to improve the quality at high compression ratios. The most promising techniques are object-based, where the images are decomposed into a number of objects based on the targeted bit rate. The encoder describes the objects in terms of contour and texture. As in the block based techniques, both intra- and inter-coding modes can be used. In the inter-coding mode, the contour and texture of the objects are motion compensated. The corresponding parameters are sent to the decoder along with the motion vectors.

The model-based approaches are not useful for general purpose scenes, but are applicable when the assumptions of a particular model are satisfied in the content, e.g., a "head-and-shoulders" scene, a "group of people" scene, graphics or computer generated scene, zooming or panning of nonmoving objects, etc.

2.3.7 Hybrid Coding

In the above sections, several image coding techniques has been discussed. Usually a combination of these techniques is used in practice to achieve a high compression ratio. For example, a combination of vector quantization and wavelet transform provides a very good coding performance [84].

Hybrid transform/predictive coding is a combination of transform and predictive coding. The configuration of this hybrid coding is depicted in Fig. 2.7, where a transform operation is followed by a predictive coder for each transform coefficients. Typically, one or two dimensional transform is applied on subimages, the coefficients are then coded using the

previous coefficients as a prediction estimate. This hybrid coding performance is in between transform and predictive coding.

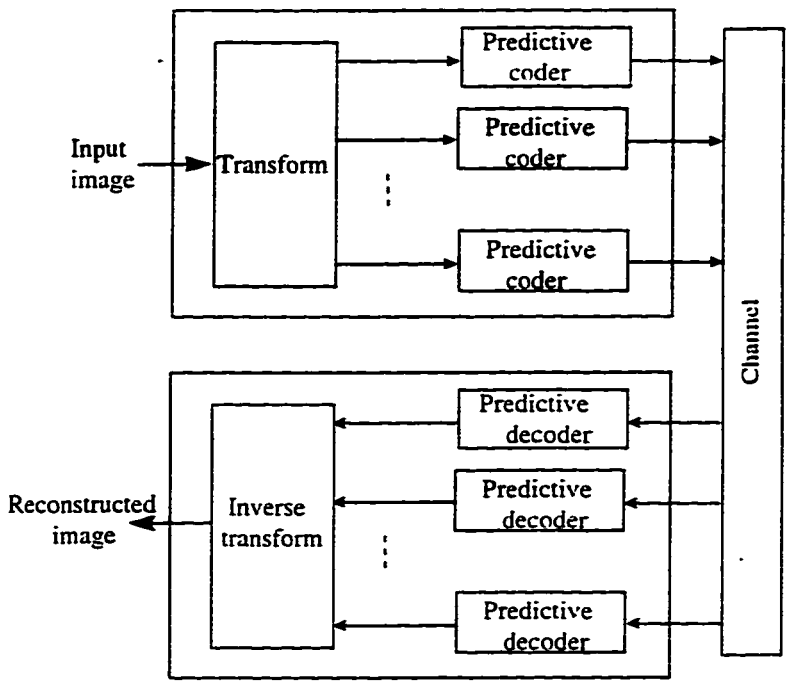


Fig. 2.7. Hybrid transform/predictive coding system

We note the choice of a compression technique depends on its computational complexity and coding performance.

2.3.8 A Comparison of Image Coding Schemes

A number of coding techniques including predictive coding, transform coding, sub-band coding, VQ and fractal coding, etc. have been reviewed. Each coding scheme posses relative advantages and disadvantages.

Predictive coding techniques have lower computational complexity and memory requirements. However, it is much more sensitive to reconstruction distortions than transform coding, since the distortions are locally distributed in predictive coding, while they are distributed over the entire image in transform coding. Hence, transform coding outperforms predictive coding at high compression ratios [30]. For one-dimensional sequences, the predictive coding performance is theoretically close to transform coding at low distortion levels. For two-dimensions, predictive coding is quite sensitive to changes in the statistics of the data. Therefore, in this case, only adaptive predictive coding schemes can achieve a compression efficiency close to transform coding. However, transform coding has higher computational complexity and memory requirements. Hybrid coding which combines transform coding and predictive coding has the advantages of good performance as well as simplicity.

Wavelet/sub-band coding uses a filterband structure and achieves a good coding performance. It achieves better subjective quality of the reconstructed images than block based transform coding.

VQ and fractal coding achieves high compression, but at the expense of high computational complexity. However, they are highly asymmetric, since compression is quite time consuming, while decompression is fast.

We note that fractal image compression has similarity with VQ in a sense that an IFS matches a domain block with a range block, while VQ matches a codeword with a data vector. However, some notable differences exist: (i) in VQ, decoding is implemented using table look-up techniques, while in an IFS, domain block is paired with a range block under an affine transformation. (ii) in VQ, the codebook is stored apart from the image being coded, while in

an IFS. the codebook is called "virtual codebook", in that it emerges during iteration. (iii) in VQ, the codebook is shared among many images, while in an IFS, the virtual codebook is specific to each image. In summary, fractal transform is a form of VQ, which employs a virtual codebook.

The object/model based coding techniques are very promising for very low bit rate coding. They are still under development and require further research.

2.4 Image Compression Standard

Recently, the Joint Photographic Experts Group has proposed the JPEG standard for still image compression. The JPEG standard [1, 2, 49] not only provides a framework for high quality compression and reconstruction of continuous-tone grayscale and color images, but also specifies the techniques for scalable image compression. In this standard, there are four modes: baseline sequential (DCT-based), hierarchical (spatial scalability), progressive (SNR scalability) and lossless modes.

The baseline sequential mode provides the basic feature of lossy image compression. The hierarchical mode reconstructs the image in multiple resolutions, where the reconstructed image is zoomed progressively through several passes. The progressive mode provides the ability of reconstructing an image in progressively improving qualities. The hierarchical and progressive modes are scalable coding techniques, which are suitable for applications in a multi-user environment. Usually, they are modified versions of the baseline algorithm. The lossless mode reconstructs the exact image by employing DPCM technique followed by entropy coding. This algorithm is ideal for applications requiring high image quality. We now

present the JPEG baseline mode coding algorithm, which is the basis for scalable coding algorithms. The scalable coding modes will be presented in chapter 3.

Baseline Sequential Mode

We illustrate the JPEG baseline coding procedure in Fig. 2.8, which includes DCT transform, DCT coefficient quantization and entropy coding.

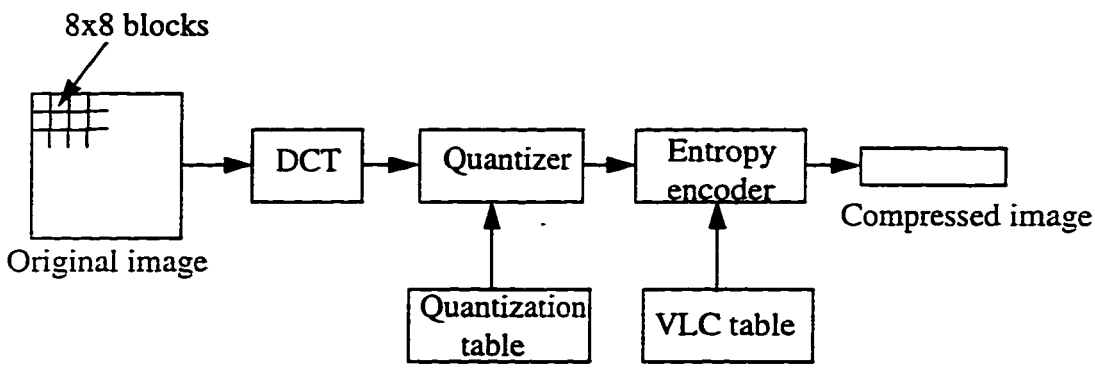


Fig. 2.8. JPEG baseline sequential encoder

The algorithm first partitions the original image into non-overlapping blocks of 8x8 pixels. Each block is then transformed using DCT into an array of 8x8 coefficients. Similar to equation (2.6), the 8x8 block DCT is defined in equation (2.10).

$$F(u,v) = \frac{1}{4}C(u)C(v)\sum_{i=0}^{7}\sum_{j=0}^{7} f(i,j)\cos\frac{(2i+1)u\pi}{16}\cos\frac{(2j+1)v\pi}{16} \qquad (2.10)$$

for $u,v = 0,1,\cdots,7$

where $$C(x) = \begin{cases} \frac{1}{\sqrt{2}} & x = 0 \\ 1 & \text{otherwise} \end{cases}$$

DCT compacts most of the perceptually important information into the DC and a few (low frequency) AC coefficients, where the DC coefficient corresponds to the mean of the pixels in an image block. The other higher frequency coefficients contain relatively less crucial details. All the coefficients are quantized using a uniform quantizer. Since the human visual system is not sensitive to degradation at high frequencies, large quantizer step sizes can be used for high frequency coefficients, while smaller sizes for low frequency coefficients. As shown in equation (2.11), the value of the coefficients is divided by the quantization step size and rounded to the nearest integer to produce the quantized coefficients. A sample quantization table has been suggested in the standard. Quantization is a lossy process. Hence most compression can be achieved by this operation.

$$C(u,v) = \left\lfloor \frac{F(u,v) + Q(u,v)/2}{Q(u,v)} \right\rfloor \tag{2.11}$$

where $Q(u,v)$ = quantization step size for coefficient (u,v)

$C(u,v)$ = rounded value of the quantized coefficient

The DC coefficient of the current block has high degree of correlation with the DC coefficient of the preceding block, which is exploited by a lossless DPCM coding technique to improve compression. The AC coefficients are not as well correlated, and therefore are coded independently. All the AC coefficients of each block are first reordered in a 1-D sequence using a zig-zag scan as shown in Fig. 2.9, and then coded using a run-length and level technique. The length of a run is the number of zero quantized coefficients skipped over, and the level is the non-zero AC coefficient's magnitude. Since quantization results in sparse matrix of block DCT, long runs are expected in most image blocks. Finally, entropy coding (Huffman or arithmetic) techniques are used to code the difference DC values and the run-lengths and levels

of the AC coefficients. Using a zig-zag scan rather than a raster scan is more efficient as it gives fewer runs and can be coded with shorter variable length codes.

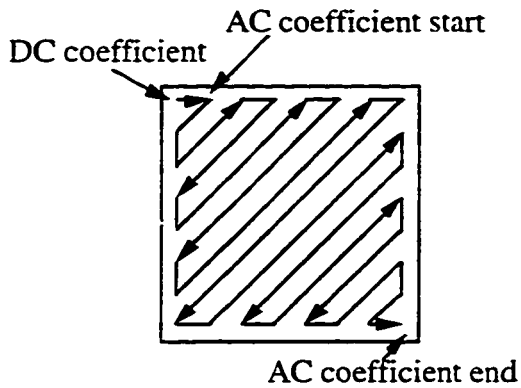


Fig. 2.9. Zig-zag scanning of block DCT coefficients

The image is reconstructed by performing the decoding operations, which is in the reverse order of encoding. In other words, the decoder performs entropy decoding, inverse zig-zag scan to recover the DCT ceofficients natural order in a block, de-quantization, and inverse discrete cosine transform (IDCT) to transform the coefficients back to spatial domain. In the JPEG standard, the baseline sequential algorithm can reconstruct the image in its original size at a specific quality.

2.5 Interframe Techniques

A video sequence consists of spatial and temporal redundancies. Therefore both intraframe and interframe techniques are needed to exploit the two kinds of redundancies. Interframe techniques exploit the correlations in the temporal dimension. In general, after the temporal redundancy is eliminated by using interframe techniques, intraframe techniques are employed

to achieve further spatial compression. Compared to intraframe techniques, interframe techniques are more complicated, since they require memory in the codec to store neighboring frames, and involve processing of information from multiple frames during a single frame interval. In this section, we present a review of the following interframe techniques: conditional replenishment, adaptive predictive coding, predictive coding with motion compensation and interframe hybrid coding.

2.5.1 Conditional Replenishment

Conditional replenishment technique [30, 40] is based on detection of the moving areas in a frame and dividing each frame into stationary and non-stationary parts, where only the changed part is coded. Let $u(m,n,i)$ represents the pixel at location (m,n) in frame i, and the predictive value is the reconstructed value $u'(m,n,i-1)$ at the same spatial location in the previous frame $i-1$. The interframe prediction error is calculated using equation (2.12).

$$e(m,n,i) = u(m,n,i) - u'(m,n,i-1) \tag{2.12}$$

Whenever the magnitude of $e(m,n,i)$ exceeds a pre-specified threshold, it is quantized and coded for transmission. At the receiver, if the pixel is from the stationary part, it is reconstructed by repeating the same location pixel from the previous frame, otherwise it is from the non-stationary part and replenished by the decoded prediction error. The reconstruction of a pixel can be obtained using equation (2.13).

$$u'(m,n,i) = \begin{cases} u'(m,n,i-1) + e(m,n,i), & \text{if } |e(m,n,i)| > \text{threshold} \\ u'(m,n,i-1), & \text{otherwise} \end{cases} \tag{2.13}$$

For transmission, codewords representing the quantized values and their addresses are generated. Isolated points or very small clusters of moving areas are ignored to make the address coding scheme efficient. There are several techniques for improvement of the basic conditional replenishment coding. For example, instead of coding the address of each changed pixel separately, the addresses and quantized values are coded in clusters along a line [41]. Another technique is to change the temporal resolution in stationary part and the spatial resolution in the non-stationary part in order to achieve further compression without affecting the subjective quality.

2.5.2 Adaptive Predictive Coding

In interframe predictive coding techniques, adaptation to motion characteristics can achieve considerable compression for a video sequence [30]. To encode a pixel, it is classified as belonging to an area of stationary (C_S), moderate/slow (C_M), or rapid (C_R) motion. The classification is evaluated by an activity index $\alpha(m,n,i)$, which is the absolute sum of interframe differences of a neighborhood N of previously encoded pixels as shown in equation (2.14). A large value of $\alpha(m,n,i)$ indicates high motion in the neighborhood N of the current pixel. The predicted value $u'(m,n,i)$ of the current pixel can be derived as shown in equation (2.15).

After predicting of current pixel value, the prediction error is then quantized, coded and transmitted to the receiver. The number of quantizer levels used for each class is proportional to its activity. This coding approach can achieve better compression than the conditional replenishment technique with approximately the same reconstructed image quality.

$$\alpha(m,n,i) = \sum_{(x,y)\in N} |u'(m+x,n+y,,i) - u'(m+x,n+y,i-1)| \tag{2.14}$$

where $N = \{(0-1),(-1,-1),(-1,0),(-1,1)\}$

$$u'(m,n,i) = \begin{cases} u'(m,n,i-1), & (m,n) \in C_S \\ u'(m-p,n-q,i-1), & (m,n) \in C_M \\ \rho_1 u'(m,n-1,i) + \rho_2 u'(m-1,n,i) - \rho_1\rho_2 u'(m-1,n-1,i), & (m,n) \in C_R \end{cases} \tag{2.15}$$

where ρ_1 and ρ_2 are the one-pixel correlation coefficients along directions m and n, respectively.

p and q are displacements, which are obtained by estimating the average displacement of the neighborhood N that gives the minimum activity.

2.5.3 Predictive Coding with Motion Compensation

Motion estimation/compensation techniques are quite efficient for exploiting the temporal redundancy in a video sequence [43, 44]. It results in considerable improvement in the compression performance compared to simple interframe techniques such as conditional replenishment and adaptive predictive coding. Therefore, it is widely used as an important interframe technique. Typically, the motion of objects in a scene can be approximated by piecewise displacement from frame to frame [30]. Motion estimation attempts to find the motion information between the current frame F_c and previous frame F_p at the transmitter, which is then used at the receiver to predict the F_c from F_p resulting in a motion compensated frame.

Two mainstream approaches for motion estimation techniques namely pel-recursive [24, 45, 46] and block-matching [43] have been proposed in the literature. Pel-recursive algorithms calculate the displacement of each pixel individually thus requiring computationally expensive

operations. These techniques do not require the transmission of motion information to the receiver since they recursively use the relative luminance change to find the motion in a frame. On the other hand, the block-matching algorithms are based on the block motion, which assume that all pixels within a block have the same motion activity [43]. These techniques are commonly used due to its reduction in computations and improvement in the prediction performance.

In the block matching process (Fig. 2.10), the current frame is divided into non-overlapping blocks of size nxn. For each block in the current frame, the best match block in the previous frame is sought within a search area using a certain search strategy. Let the search range be $+p/-p$ in both the horizontal and the vertical directions, then the search area consists of $(n+2p)^2$ pixels, which is centered around the candidate block with identical coordinates as the block in the current frame. The motion specified by of a two-dimensional motion vector provides the relative displacement between the current block and the best matched region in the previous frame.

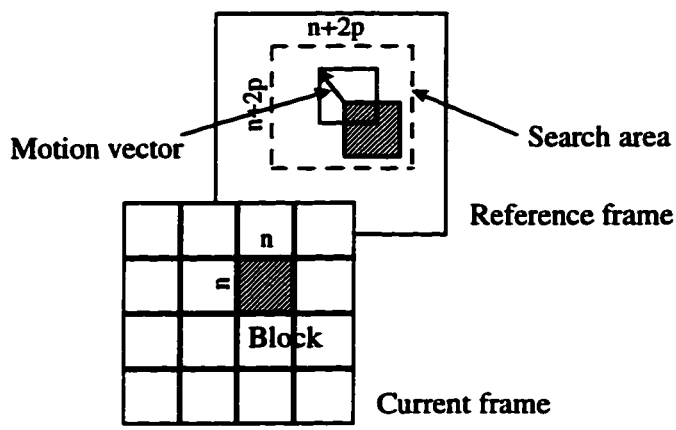


Fig. 2.10. Block matching process for motion estimation

The most intuitive search strategy for block matching is the full search algorithm (FSA). In order to find the best match for a block in the current frame, all possible displacements within the chosen search range are evaluated using the block matching criterion. Several block matching criteria are available [42], such as the mean square error (MSE) and the mean absolute error (MAE). The candidate block which minimizes the error is selected as the best match. Since the MAE criterion has lower computational complexity, it is generally used. The MAE criterion is given by equation (2.16).

$$MAE(\Delta x, \Delta y) = \sum_{i=1}^{n} \sum_{j=1}^{n} |S(i+\Delta x, j+\Delta y) - R(i,j)| \tag{2.16}$$

where $R(i,j)$ is a block's pixel in the current frame

$S(i+\Delta x, j+\Delta y)$ is a candidate block's pixel in the previous frame within the search area

We note that, in the FSA algorithm, all possible $(n+2p)^2$ candidate blocks are searched, which involves a high computational complexity. To reduce the computational burden, several fast algorithms have been proposed [23, 47, 48]. These algorithms rely on the assumptions that the error (MSE or MAE) increases monotonically as the search moves away from the optimal vector. Therefore the computations are reduced by limiting the number of candidate blocks.

The basic idea of motion estimation/compensation is similar to adaptive predictive coding, where the prediction is the best matched block data from the previous frame [34]. High compression can be achieved by coding the prediction error image (the difference between the current and motion compensated frames) and motion vectors. The accuracy and speed of motion estimation determine the success of the motion-compensated coder.

2.5.4 Interframe Hybrid Coding

Interframe hybrid coding is a combination of transform and interframe prediction coding [44]. We note that the motion estimation/compensation can be considered as a adaptive predictive coding scheme (section 2.5.3). Typically, each frame is partitioned into two-dimensional non-overlapping blocks. For each block in the current frame, the prediction error can be obtained by subtracting the data of the best match in the previous frame from the current block data. Generally speaking, most intraframe coding techniques can be applied to encode the prediction errors for further spatial compression [34]. Among them, transform coding techniques such as DCT is widely used.

Recall from section 2.3.2, in transform coding systems, the transform coefficients are first visually quantized to achieve compression without degrading the subjective quality. The quantized coefficients are then entropy coded for further compression. Finally, the coded coefficients are transmitted to the receiver along with the motion vectors. A typical interframe hybrid coder is shown in Fig. 2.11.

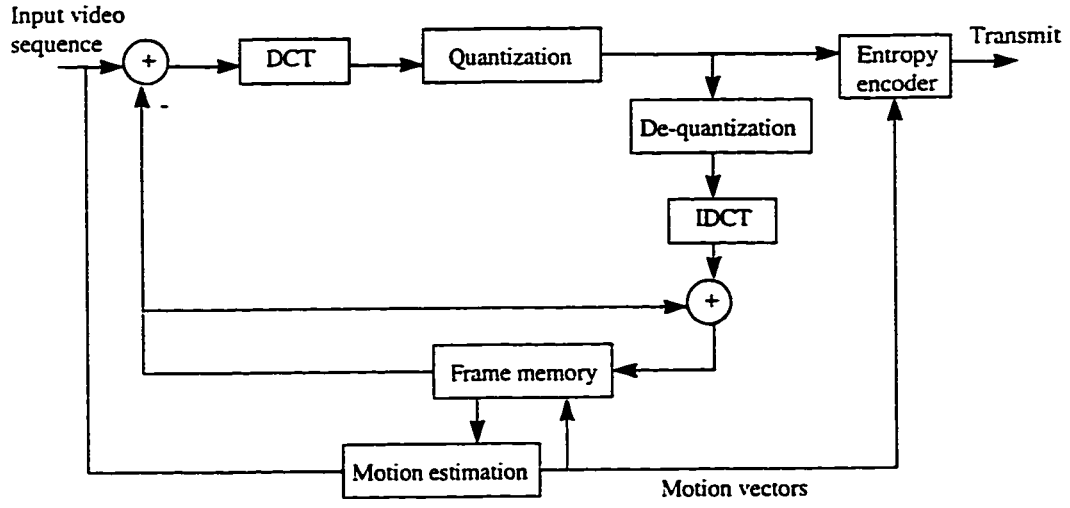


Fig. 2.11. Interframe hybrid coding with motion compensation

2.6 Video Compression Standards

The International Telecommunications Union (ITU) and the International Standards Organization (ISO) have proposed standards for video compression known as the H.261 and MPEG standards, respectively. The H.261 standard [5, 57] is aimed at video phone and video conferencing applications, where real-time encoding and decoding are essential. It is often called the $p \times 64$ standard since the targeting bit rate is $p \times 64$ kbits/s, where p is in the range of 1 to 30. ITU-T has recently developed the international draft standard H.263 [58, 59], which is closely related to the well known H.261. H.263 provides the same subjective image quality at less than half the bit rate [60]. It employs an improved version of the classical block-based motion-compensated DCT (hybrid) coding used in H.261 and MPEG standards [62].

The MPEG standard [3, 4, 56, 63] addresses motion video as well as audio coding according to the ISO/IEC standardization process. MPEG-1 [3, 56] specifies a coded representation that can be used for compressing video sequences up to bit rate around 1.5 Mbits/s. It was developed in response to the growing need for a common format for representing compressed video on various digital storage media such as CD's. DATs (digital audio tape), and optical drives. The use of MPEG-1 means that motion video can be manipulated as a form of computer data and can be transmitted and received over existing and future networks. Further developments in the area of video coding techniques are based on a target rate of up to 100 Mbits/s. This is known as MPEG-2 [4]. It strives for a higher resolution similar to the digital video studio standard CCIR601 and leading to HDTV. The MPEG-2 video standard specifies the coded bit stream for high-quality digital video. As a compatible extension, MPEG-2 video builds upon the completed MPEG-1 standard by supporting

interlaced video formats and a number of advanced features including those supporting HDTV. The MPEG-2 Main Profile was defined to support digital video transmission in the range of about 2 to 80 Mbits/sec over cable satellite and other broadcast channels as well as for digital storage media and other communication applications [61].

Very low bit rate high compression video coding is becoming popular. ISO has initiated the MPEG-4 standard activity [61, 63, 64, 82, 83] to investigate novel algorithms for high compression coding in mobile multimedia applications based on techniques such as wavelet, vector quantization, fractal, object and model based coding. In addition to compression efficiency, MPEG-4 is targeted to be flexible, extensible, and providing object-based interactivity. It will support new ways of communication, universal access and manipulation of digital audio-visual information. MPEG-4 standardization is still in progress and stated to have a draft international standard by 1998.

We now discuss the MPEG-1 and MPEG-2 algorithms in detail, since the techniques proposed in this thesis implement encoding of scaled MPEG video in the compressed domain.

2.6.1 Overview of the MPEG Algorithm

MPEG coder is essentially an interframe hybrid coder, which uses DCT and motion estimation/compensation techniques to exploit the spatial redundancy within a frame and temporal redundancy between the successive frames in a video sequence, respectively. It achieves a high compression ratio as well as a good picture quality with the requirement to have the possibility of random access to the coded bit stream. Intraframe coding best meets the random access requirement, while interframe coding can achieve high compression. This

requires a careful balance between intra and interframe coding. Three main picture types are defined in MPEG.

Intra coded pictures (I-pictures) are coded without reference to other pictures, which are treated as still images. They provide access points to the coded sequence where decoding can begin, but are coded with only moderate compression. Predictive coded pictures (P-pictures) are coded more efficiently using motion compensated prediction from a past I-picture or P-picture, and are generally used as a reference for further prediction. Bidirectionally-predictive coded pictures (B-pictures) provide the highest degree of compression but require both past and future reference pictures for motion compensation. B-pictures are never used as references for prediction. In MPEG, a video sequence is organized into a set of group of pictures (GOP), which is usually a combination of I, P and B-pictures. A GOP must have at least one I-picture. The number and organization of the three picture types in a GOP (GOP pattern) is very flexible, which depends on the requirements of the applications and is defined by the encoder. For example, a GOP pattern IBBPBBPBBP and the temporal prediction for P and B-pictures are shown in Fig. 2.12. It can be seen that the MPEG coder contrasts with the simple interframe coder, where only the first frame is intra-coded, while the rest are inter-coded.

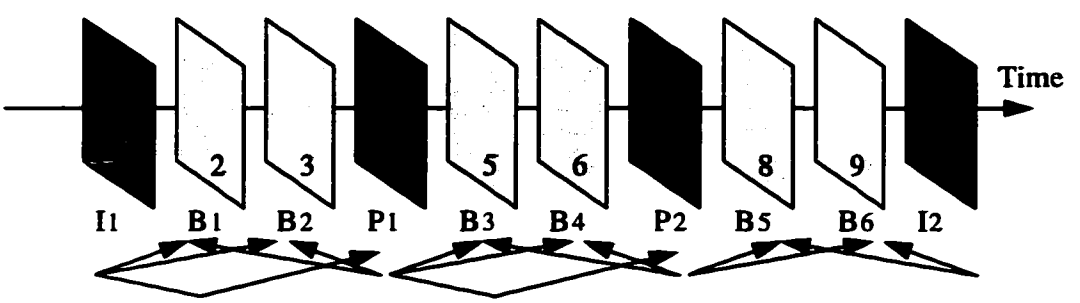


Fig. 2.12. Prediction in MPEG video sequence

MPEG defines a 16x16 region as a macroblock, which is the basis for motion estimation/compensation. The forward motion estimation technique (as discussed in section 2.5.3) is used for causal prediction of a P-picture from a previous picture. In addition, bidirectional motion estimation is employed for non-causal interpolated prediction of a B-picture from past and future pictures. This results in an improved performance compared to unidirectional motion estimation. We note that all macroblocks in I-pictures are intra-coded, while the macroblcks in P and B-pictures can be either inter-coded or intra-coded depending on the effectiveness of motion estimation.

Both the original data for an intra-coded macroblock and the prediction error for an inter-coded macroblock have a high spatial redundancy. These macroblocks are split into blocks of size 8x8 pixels and transformed into the DCT domain to remove the spatial correlations. The DCT coefficients are visually weighted and quantized in an irreversible manner that discards the less important information. MPEG uses different quantization tables for intra- and inter-coded macroblocks. The step sizes of the quantization matrix can be modified using the quantizer scale factor, which may be changed between macroblocks. This makes possible control on the quantization within a frame on a macroblock basis. After quantization, the coefficients are reorganized in a zig-zag scan order. The DC coefficients for intra-coded blocks are compressed using a DPCM technique, while the DC coefficients for the inter-coded blocks and all the AC coefficients are coded using two-dimensional run-length and variable length coding. In addition, for inter coded macroblocks, the associated motion vectors are encoded differentially with respect to the last transmitted motion vector, using variable length codes. We note that the transformation, quantization, zig-zag scan, run-length and variable length coding techniques for the picture blocks are similar to those defined in the JPEG standard.

2.6.2 Motion Compensation and Macroblock Type Selection

In the MPEG standard, after the motion estimation process for macroblocks in P- and B-pictures, an encoder will make a series of decisions of choosing between the different types of macroblock. An encoder uses the following order for the decisions:

(i) Motion compensation or no motion compensation, i.e. is a motion vector transmitted or is it assumed to be zero.

(ii) Inter- (non-intra) or intra-coding, i.e. is the macroblock predicted using the motion vector found in step (i), or is it intra-coded.

(iii) If the macroblock is non-intra coded, it decides whether the prediction error large enough to be coded using DCT.

(iv) It decides whether the quantizer scale is satisfactory or should be changed.

In this thesis, we emphasize on the inter/intra coding decision (step (ii)), which is related to our proposed approach for re-encoding scaled MPEG video which will be presented in chapter 5.

For P- and B-pictures, intra-coded macroblocks are possible whenever the macroblocks cannot be effectively predicted from the reference pictures. For example, a scene change or forced cut may appear between successive pictures, or the motion activity is too high to be captured within the search area for motion estimation.

An intuitive way to make the decision is to compare the total bit numbers required to code the same macroblock as inter- or intra-coded. The method using fewer bits may be chosen. However, this is a computationally expensive operation. In the MPEG standard, the inter/intra coding decision is made based on the variance of the luminance component of a macroblock. The variances of the current macroblock V_c and of the difference macroblock V_d are compared.

The difference macroblock is obtained by subtracting the motion-compensated previous macroblock from the current macroblock.

We use equation (2.17) and (2.18) to calculate variance V_c and V_d, respectively. Note that, when calculating the variance of the difference macroblock, the average value is assumed to be zero.

$$V_c = \frac{1}{256}\sum_{i,j=0}^{15}(x_{ij})^2 - (\frac{1}{256}\sum_{i,j=0}^{15}x_{ij})^2 \tag{2.17}$$

$$V_d = \frac{1}{256}\sum_{i,j=0}^{15}(x_{ij} - y_{ij})^2 \tag{2.18}$$

where x_{ij}, y_{ij} are values of pixel (i, j) in the current macroblock and the motion-compensated previous macroblocks, respectively.

The macroblock type decision for P-pictures is based on Fig. 2.13. It can be seen that when V_d is less than 64 or V_d is less than V_c, inter coding is chosen.

The coding decision for B-pictures is more complicated than P-pictures. Since B-pictures are coded using bidirectional prediction, a macroblock may be motion compensated in three different manners: forward predictive coded, backward predictive coded and bidirectionally predictive coded (interpolated).

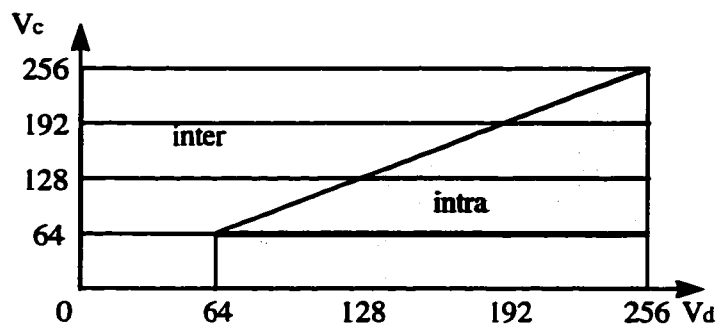


Fig. 2.13. Inter/intra coding decision making for P-pictures

For the forward or backward motion compensation, the motion-compensated macroblock is constructed from the previous reference picture or the future reference picture, where only one motion vector is associated with each macroblock. For the interpolated motion compensation, the interpolated motion-compensated macroblock is obtained by averaging two motion compensated macroblocks from both the previous and future reference pictures, and two motion vectors are required to be transmitted to the decoder.

Using a similar technique as for P-pictures, an encoder decides whether to code a macroblock in B-pictures as inter coded using the best of the three possible prediction (forward or backward or interpolated) or to code it as intra coded by calculating and comparing the variances of the difference macroblock and current macroblock. The macroblock type with the smallest variance is chosen (Fig. 2.14). If the two variances are equal, inter-coding is chosen.

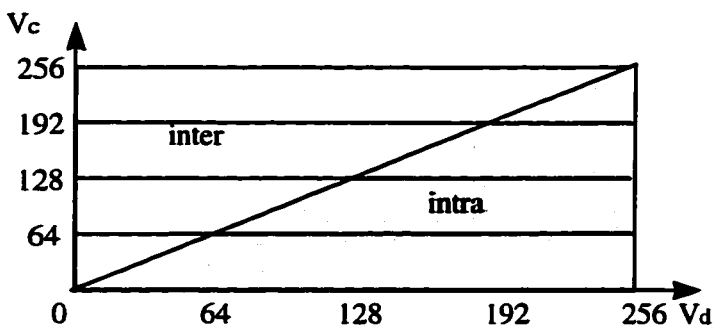


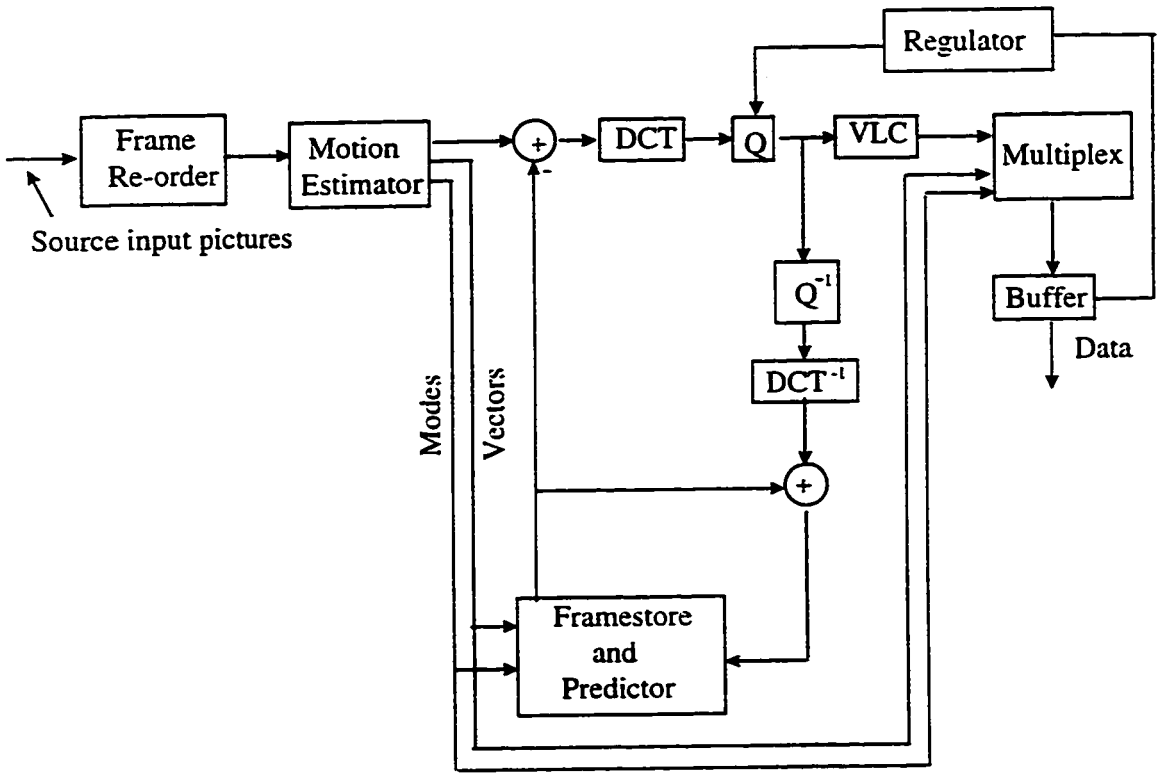
Fig. 2.14. Inter/intra coding decision making for B-pictures

It can be seen that in P- and B-pictures, each macroblock can be one of a number of different types. An encoder stores these macroblcok type information with the prediction error signal in each macroblock, which will be used for decoding.

2.6.3 MPEG Codec

MPEG Encoder

MPEG does not specify an encoding process. It specifies the syntax and semantics of the bitstream and the signal processing in the decoder. As a result, many options are left open to encoders to trade-off cost and speed against picture quality and coding efficiency. Fig. 2.15 shows the main functional blocks of the MPEG encoder.



where DCT is the discrete cosine transform, DCT^{-1} is the inverse discrete cosine transform
Q is the quantization, Q^{-1} is the inverse quantization
VLC is the variable length coding

Fig. 2.15. Simplified MPEG video encoder block diagram

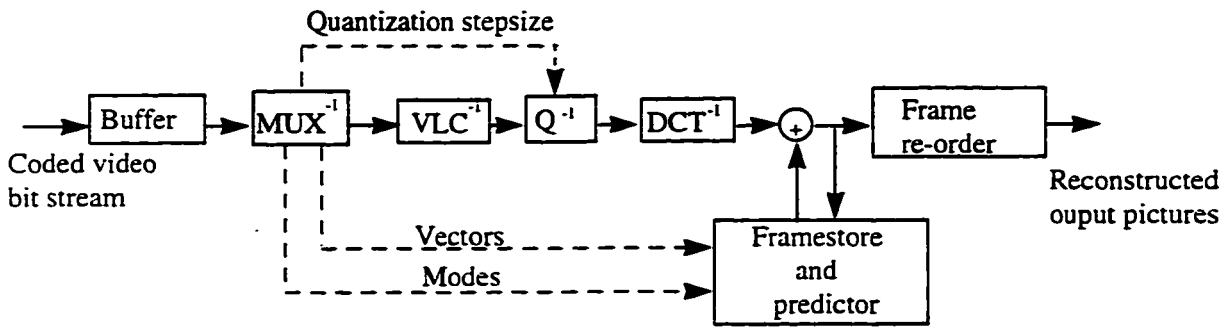
The input video signal must be digitized and represented as a luminance and two colour difference signals (Y, Cr, Cb). This may be followed by preprocessing and format conversion

to select an appropriate window, resolution and input format. After the preprocessing of the input pictures, the encoder defines the GOP pattern, and performs motion estimation for each 16 by 16 macroblock in the picture. Typically, in P-pictures, one vector is needed for each macroblock and in B-pictures one or two vectors are needed.

If B-pictures are used, some reordering of the picture sequence is necessary before encoding. Since B-pictures are coded using bidirectional motion compensated prediction, they can be decoded only after the subsequent reference (an I- or P-picture) has been decoded. Hence, the pictures are reordered by the encoder so that the pictures arrive at the decoder in the appropriate order for decoding. The correct display order is recovered by the decoder.

MPEG Decoder

Decoding is the inverse of the encoding operation. It is considerably simpler than encoding as there is no need to perform motion estimation and there are many fewer options. The decoding process is defined by the MPEG standard. Fig. 2.16 shows the main functional blocks of the MPEG decoder.



where DCT^{-1} is inverse discrete cosine transform
Q^{-1} is inverse quantization
MUX^{-1} is demultiplexing
VLC^{-1} is the variable length decoder

Fig. 2.16. Basic MPEG video decoder block diagram

In the decoder, if an I-picture or a P-picture is reconstructed, it is stored as a reference picture for subsequent pictures. Before the pictures are displayed, they may need to be re-ordered from the coding order to their natural display order.

2.6.4 Structure of MPEG Bit Stream

The MPEG simulation model is organized into six layers (Fig. 2.17), each of which either supports a signal processing functionality or a system function.

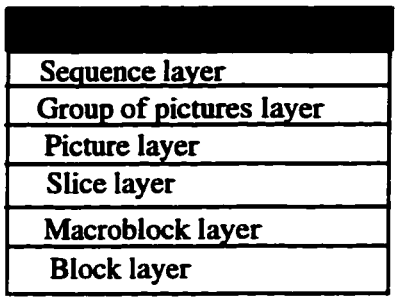


Fig. 2.17. Six layers in MPEG simulation model

Sequence layer

A coded video sequence commences with a sequence header and is followed by one or more groups of pictures and is terminated by a sequence end code. There may be a sequence header immediately preceding each of the groups of pictures. Within each sequence, pictures are decoded continuously.

Group of pictures layer

A group of pictures is a series of one or more consecutive pictures intended to assist random access into the sequence. In the coded bit stream, the first coded picture in a group of pictures is an I-picture. The order of the pictures in the coded bit stream is the order in which the decoder processes them. In display order, the last picture in a group of pictures is always an

I-picture or a P-picture, and the first is either an I-picture or the first B-picture of the consecutive series of B-pictures which immediately precedes the first I-picture (Fig. 2.18).

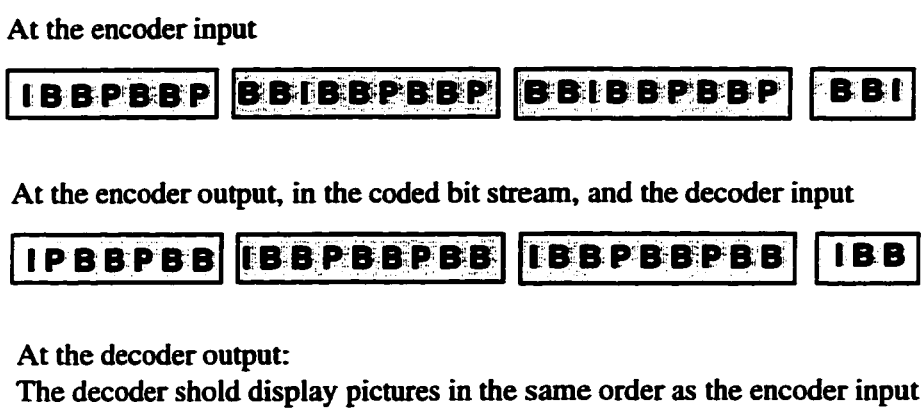


Fig. 2. 18. The order of the pictures in a GOP

Picture layer

A picture in MPEG terminology is the basic unit of display and corresponds to a single frame in a video sequence. The spatial dimensions of a frame are variable and are determined by the requirements of an application.

Slice layer

A slice is the basic processing unit in the MPEG coding scheme, which is a series of an arbitrary number of macroblocks in a picture by raster-scan order. A slice is an autonomous unit since coding a slice is done independently from its neighbors.

Macroblock layer

A macroblock of size 16x16 is the basic coding unit in the MPEG algorithm. Since MPEG requires that the color difference signals (Cr and Cb) be sub-sampled with respect to the luminance by 2:1 in both the vertical and horizontal directions, a macroblock contains a section

of the luminance component and the spatially corresponding chrominance components (Fig. 2.19).

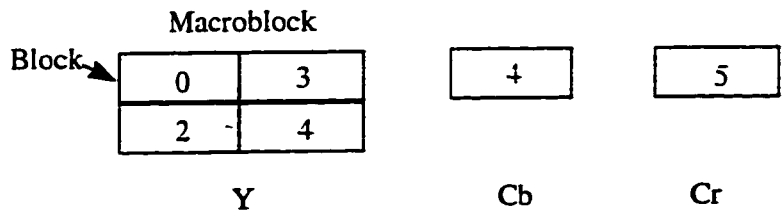


Fig. 2.19. Macroblock structure

Block layer

A block of size 8x8 is the smallest coding unit in the MPEG algorithm, which is the unit for DCT. A block can be one of three types: luminance (Y), red chrominance (Cr), and blue chrominance (Cb).

2.7 Summary

In this chapter, we first present the concepts of entropy (lossless) coding, source (lossy) coding and hybrid coding are presented. This is followed by a review of lossless data compression techniques including run-length coding, Huffman coding and Arithmetic coding. Then the intraframe coding techniques including predictive coding, transform coding, sub-band coding and vector quantization, fractal image compression, etc., are reviewed, followed by the JPEG standard for still image compression. Next, we review the interframe coding techniques. such as conditional replenishment, adaptive predictive coding, predictive coding with motion compensation and interframe hybrid coding. Finally, we present a brief review of video compression standards including H.261, H.263, MPEG-1, MPEG-2 and MPEG-4. A particular emphasis is placed on the review of MPEG algorithm.

Chapter 3

Review of Scalable Image/Video Compression

In this chapter, we first review the basic concepts of scalable image/video compression including SNR scalability, spatial scalability and temporal scalability. This is followed by a brief review of spatial domain and transform domain coding techniques for SNR and spatial scalability. In section 3.2, we present the JPEG scalability coding modes, namely progressive (SNR scalability) and hierarchical (spatial scalability). Finally, the scalable features in the MPEG-2 standard, namely SNR scalability, spatial scalability and temporal scalability are discussed in section 3.3.

3.1 Review of Scalability

Image and video applications are often targeted to a multi-user environment where the data may be transmitted and accessed by a large number of display devices, each with a different

spatial. temporal or quality requirement. storage and processing capabilities. For example. in many interactive visual applications, one of the key required features is the capability of browsing large visual databases to locate and retrieve the data of interest. The visual data may be first displayed at a low spatial resolution or picture quality, which is progressively enhanced to increasing sizes and qualities based on the viewer's request. In HDTV with embedded TV systems, the TV signals may be migrated to higher temporal resolution HDTV signal. In addition, many interactive visual communication applications involve transmission of images and video over low speed, low bandwidth channels or a dual-priority communications network, where the low priority data transmission depends on the availability of bandwidth. Hence there is a need for scalable visual compression techniques which represent visual data with multiple spatial, temporal or quality resolutions [6, 7, 9. 10, 68]. Scalability is extremely important in applications such as browsing image catalogues, querying multimedia databases, and interactive multimedia communications.

Scalability is usually achieved by coding the visual data into two or more layers, where the lowest layer provides the basic reconstruction of visual data, and the higher layer (enhancement layer) is coded to enhance the information provided by the lower layer. The different layers can be classified into high priority and low priority. This feature makes possible graceful degradation in performance for visual transmission applications over dual-priority networks in the event of lack of bandwidth or network congestion.

Generally, scalability can be classified into three categories: SNR scalability, spatial scalability and temporal scalability. A combination of these basic scalabilities is referred to as hybrid scalability.

3.1.1 SNR Scalability

SNR scalability is a generic feature referring to the representation of images in different qualities or SNR at a fixed size. Sometimes it is referred to as progressive image transmission (PIT) [8, 69, 70]. SNR scalability is mainly targeted toward applications which require graceful degradation in performance in the event of constrained bandwidth or computing resources.

A quick initial approximation image is constructed at very low bit rate. This image is progressively improved to full quality by adding more information from the bit stream in several passes. This feature can be used in image browsing interactive applications, where an image database is searched based on a particular criterion. The retrieval process can be terminated if the intermediate version of the image is of satisfactory quality or if the image is found to be of no interest.

Several techniques for obtaining progressively improving quality sequences (SNR scalability) by reorganizing the image data into a fixed number of levels in either the spatial or transform domain have been proposed in the literature [11, 14, 71, 73, 74].

The spatial domain techniques encode a image by reconstructing it progressively on the image pixel basis, such as bit plane and sub-sampling [71]. The transform domain techniques can be classified into coefficient scanning [73], bit slicing [74], and S-transform [11, 14]. In the coefficient scanning technique, a low resolution approximation is first reconstructed only from a few coefficients, then the reconstructed image is progressively improved by increasing the number of higher order coefficients. In the bit slicing technique, the full bit assignment map in the transform domain is sliced into a set of bit layers which are then progressively transmitted. The S-transform is a hierarchical approach for image representation, which can be used for

achieving both SNR and spatial scalabilities. It builds the image into a pyramid data structure by successive Hadamard transform.

In S-transform, an input image is first partitioned into non-overlapping blocks of size 2x2 pixels. Each block of pixels is then transformed into four coefficients as follows. The DC coefficient is the average of the four pixels in a block, while AC coefficients presents high frequency information.

$$\begin{aligned} d_0 &= \frac{1}{4}(x_1 + x_2 + x_3 + x_4) \\ d_1 &= \frac{1}{4}(x_1 + x_2 - x_3 - x_4) \\ d_2 &= \frac{1}{4}(x_1 - x_2 - x_3 + x_4) \\ d_3 &= \frac{1}{4}(x_1 - x_2 + x_3 - x_4) \end{aligned} \tag{3.1}$$

where d_0 = DC coefficient, d_1, d_2, d_3 = AC coefficients

$x_1, \cdots, x_4$ = pixel values in a 2x2 image block

In the first decomposition pass, a sub-image with half the resolution of the original image can be obtained by assembling the DC coefficients of all the image blocks. Similarly, sub-images can be obtained for the three AC coefficients. The DC sub-image is then further decomposed by successive application of Hadamard transform. To reconstruct the image, the smallest DC sub-image (up-sampled to the original size) is first displayed, which is then improved by adding information from the corresponding AC coefficients. Therefore, SNR scalability can be achieved by several passes until the highest SNR resolution image is reconstructed.

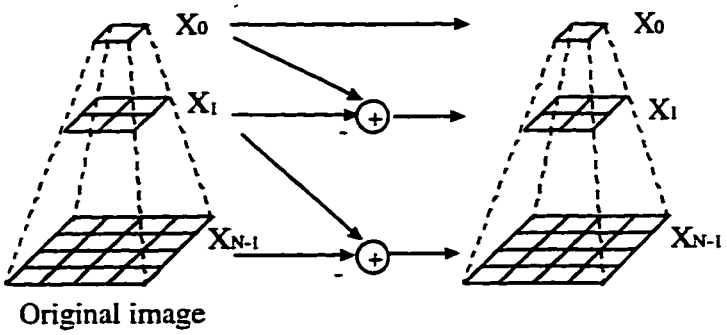
3.1.2 Spatial Scalability

Spatial scalability refers to the feature of image representation in different sizes or spatial resolutions [9, 10, 68]. It allows multi-resolution coding, and is suitable for a variety of visual services. In addition, it provides the capability of reconstructing images in lower spatial resolutions using only a small portion of the compressed bit stream. A quick iconic view of the image corresponding to the lowest resolution image stored in the database is first required. The selected image can then be zoomed to progressively increasing sizes based on user need until the full spatial resolution image is reconstructed.

Techniques for achieving spatial scalability are typically based on pyramidal data structures [6, 8, 9, 12], where the image data is organized in a hierarchical fashion, such that each level of the hierarchy corresponds to a specific spatial resolution of the image. A pyramid data structure reorganizes the input image X as a sequence of matrices $\{X_i\}$, $(i = 0,1,\cdots,N-1)$, where $\{X_{i-1}\}$ is the reduced resolution version of $\{X_i\}$. In this structure, X_0 and X_{N-1} correspond to a single pixel matrix and the original image, respectively.

The techniques proposed in the literature for spatial scalability can be classified as either spatial domain or transform domain techniques. In the spatial domain techniques, the pyramid can be formed by sub-sampling, mean filtering, Gaussian-based low pass filtering [13], etc. Usually the difference pyramid is also needed for exact reconstruction of the image. A simplest pyramid is a quadtree as shown in Fig. 3.1. For a given image of size $2^n x 2^n$, a pyramid data can be formed by successively operating on the neighboring $2x2$ pixels [8, 12, 13, 72], where the operations can be mean filtering, median filtering, etc.

Most spatial domain pyramidal techniques are often used in conjunction with the transform domain techniques to provide a good coding performance. The transform domain techniques can be implemented by S-transform or DCT on each pyramidal level.



(a) Quadtree pyramid of an input image (b) Difference quadtree pyramid

Fig. 3.1. Formation of a quadtree pyramid

We recall from section 3.1.1, that S-transform achieves both SNR and spatial scalability. It hierarchically decomposes the input image into a pyramidal representation, where the low pass sub-images represent the original image in different spatial resolution.

The other transform domain technique, i.e. the DCT domain technique has two flavors, namely spatial domain pyramid [9] and frequency domain pyramid [6]. Spatial pyramids generally use 8x8 DCT at every layer as recommended in the JPEG and MPEG standards, while frequency pyramids use DCT of smaller size in the lower layers. In a frequency pyramid, the original image is first filtered and down-sampled to form a quadtree, and then block DCT is performed on each level based on different block sizes. As shown in Fig. 3.2, for a four level quadtree, the original image in the fourth level is partitioned into non-overlapping blocks of size $8x8$, while, the block sizes in the first to the third level are $1x1$, $2x2$, and $4x4$.

respectively. For efficient compression, the DCT coefficients at each level are used to predict the respective low frequency coefficients in the next higher layer.

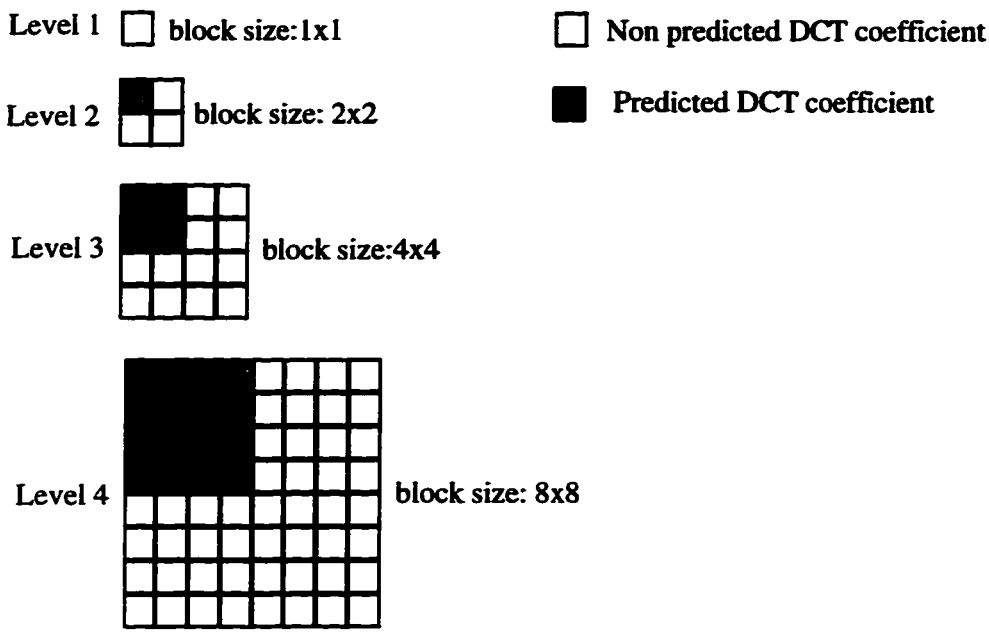


Fig. 3.2. DCT blocks for different layers in a frequency domain pyramid

We note that, in transform domain techniques, the S-transform or DCT are performed recursively at different spatial resolutions on each pyramid level. Hence, it requires down-sampling operations in the spatial domain before any transform domain techniques can be employed.

3.1.3 Temporal Scalability

Temporal scalability refers to a hierarchical representation of video such that each level of the hierarchy has a different frame rate [10, 68]. It involves partitioning of video frames into layers, where the lower temporal resolution (frame rate) can be progressively enhanced to

higher temporal resolution. Temporal scalability is only applicable to video. For a range of diverse video applications from telecommunications to HDTV, it is necessary to migrate from lower temporal resolution systems to higher temporal resolution systems.

Temporal scalability is achieved by skipping certain frames at the base layer, where the skipped frames are then encoded at the enhancement layer (Fig. 3.3). The top layer is predicted from either the decoded frame at the base layer, or from the previously decoded frame at the top layer. At the top layer, it is prohibited to use backward prediction, which avoids the need for frame reordering and reduces complexity.

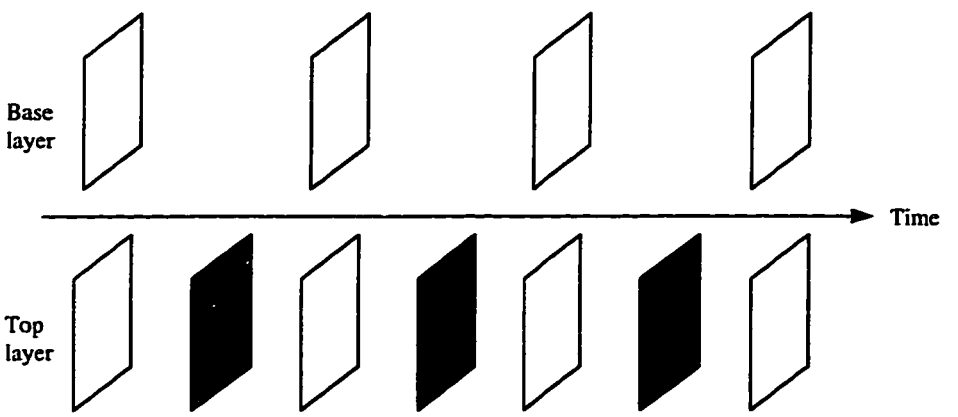


Fig. 3.3. Illustration of temporal scalability

3.2 JPEG Standard for Scalability

Recall from section 2.4 that the JPEG standard has four modes: baseline sequential (DCT-based), hierarchical (spatial scalability), progressive (SNR scalability) and lossless modes. We now present the definition of the SNR and spatial scalability techniques within the JPEG standard.

3.2.1 JPEG SNR Scalability

For SNR scalability, JPEG employs transform domain techniques. It provides two approaches, namely spectral selection and successive approximation. which correspond to the aforementioned coefficient scanning and bit slicing techniques, respectively.

In the spectral selection approach, the DCT coefficients are partitioned into a number of bands according to their frequencies, and each band is compressed and stored separately. The block DCT coefficients may be partially encoded by selecting a specified band of coefficients from the zig-zag sequence within a given scan. The first band is generally reserved for the DC coefficient. The AC coefficients are split amongst the remaining available bands. Each band is defined by a header which specifies the indices corresponding to its first and last coefficients. A coarse quality image can be obtained by decoding the block DC coefficients, which can be progressively improved by gradually decoding the corresponding AC bands, and adding them to the coarse quality image after inverse DCT (IDCT). This operation is possible due to the linearity of DCT. Hence spectral selection reconstructs the image in different qualities thereby providing SNR scalability.

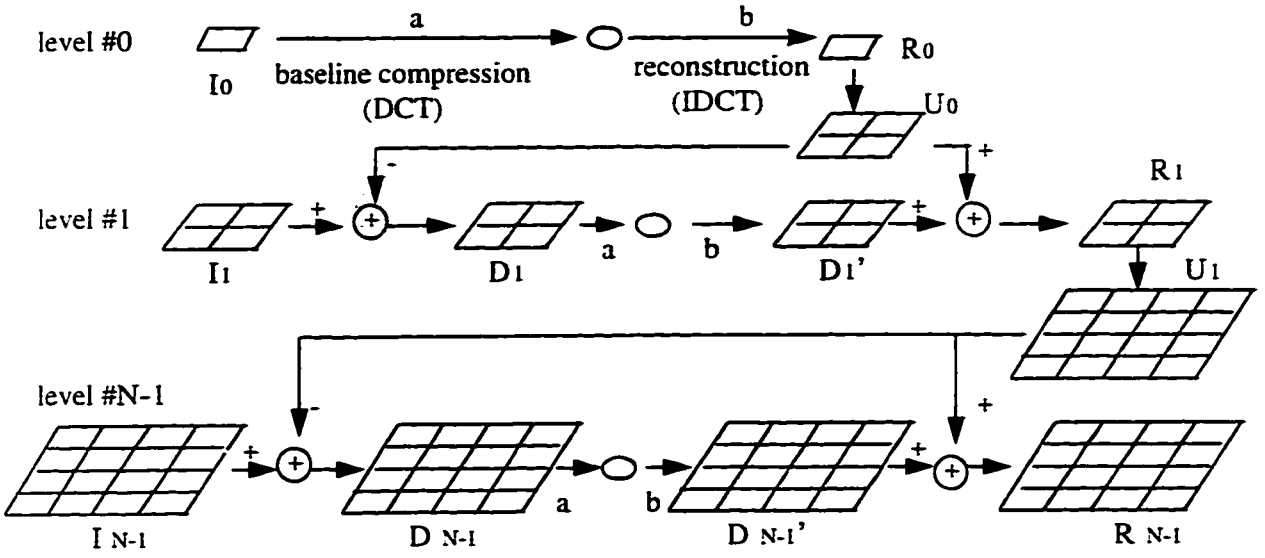
In the successive approximation approach, all the quantized DCT coefficients are sliced into a number of bit planes. The most significant bit plane has certain number of the most significant bits of the coefficients. Each lower bit plane increases the coefficient precision by 1 bit. In the first scan, the most significant bit plane is encoded at a reduced precision. It provides a first approximation coarse image with both high and low frequency information. which results in a reasonable reconstruction of the edges at low bit rates. In subsequent scans, the lower bit planes are then encoded individually to progressively improve the image quality.

We note both of these SNR scalability techniques (spectral selection and successive approximation) are based on block DCT coefficients, which are generated by using the JPEG baseline compression scheme, therefore a JPEG baseline bit stream can be easily updated to a SNR scalable bit stream without decompression. In other words, we can implement SNR scalability directly in the compressed domain in the JPEG standard.

3.2.2 JPEG Spatial Scalability

The spatial scalability technique proposed in the JPEG standards uses a combined spatial and transform domain technique. The hierarchical mode encoder employs a pyramidal structure for encoding different spatial resolutions of the image. It is based on encoding the smallest size image, and differential encoding of the larger size images for different layers using DCT.

We illustrate the spatial scalable coding procedure in Fig. 3.4. First, to prepare the data, the original image is successively filtered and down-sampled to several lower resolution images. The lowest layer image I_0 (iconic version) is DCT encoded and quantized. The resulting bit stream is decoded to reconstruct the lowest layer image R_0, which is then interpolated and up-sampled by a factor of two in both the horizontal and vertical directions to form the prediction U_0 of the original image at next higher layer. To encode the next higher layer I_1, a difference image D_1 calculated by computing pixelwise difference between I_1 and U_0, which is DCT encoded and quantized. This data is decoded to reconstruct the difference image D_1', which is added to U_0 to reconstruct the image R_1. Proceeding in a similar fashion, R_1 is up-sampled to the next higher layer image U_1. The entire procedure is repeated until the full resolution image has been encoded.



I = down sampled image, D = diff image, D' = reconst diff image,
R = reconst image, U = up sampled image

Fig. 3.4. JPEG hierarchical mode encoder

To decode the JPEG hierarchical mode bit streams, the smallest iconic image is first reconstructed as image R_0. The bit streams corresponding to the difference images D_1, D_2, ..., D_{N-1} are then decoded and added to the respective up-sampled images U_0, U_1, ..., U_{N-2} until the full resolution image R_{N-1} is reconstructed. Hence, spatial scalability is achieved by first displaying the lowest resolution image, which is gradually zoomed to progressively higher resolutions. We note that the encoder involves down-sampling and up-sampling as well as compression and decompression operations, while the decoder requires only up-sampling and decompression operations.

3.3 MPEG Standard for Scalability

Recall form section 2.6 that the MPEG-2 standard was developed as a compatible extension of MPEG-1. The MPEG-2 syntax can be divided into two major categories: non-scalable syntax and scalable syntax. The former is a super set of the syntax defined in MPEG-1, and supports interlaced video formats. The latter provides the feature of scalability, which structures the total bit stream into multiple layers, starting from a standalone base layer, and adding a number of enhancement layers. MPEG-2 offers SNR scalability, spatial scalability and temporal scalability, and also supports hybrid scalability.

3.3.1 MPEG SNR Scalability

SNR scalability involves generating two video layers of the same spatial resolution but different video qualities from a single video source, where the lower layer is coded by itself to provide the basic video quality, and the higher layer is coded to enhance the lower layer [4].

SNR scalability defines a mechanism to refine the DCT coefficients in the lower layer by using the DCT coefficients coded in the higher (enhancement) layer of a stream. As illustrated in Fig.3.5, when decoding, the final block DCT coefficients can be obtained by combining the de-quantized coefficients from two layers. This technique is similar as the JPEG SNR scalability technique. After block DCT coefficient reconstruction, the procedure for temporal prediction and frame reconstruction is identical to the respective procedure for decoding of a single layer bit stream (as discussed in section 2.6). Note that all the prediction information is contained in the lower layer. Therefore, it is not possible to reconstruct an enhancement layer without decoding the lower layer data before.

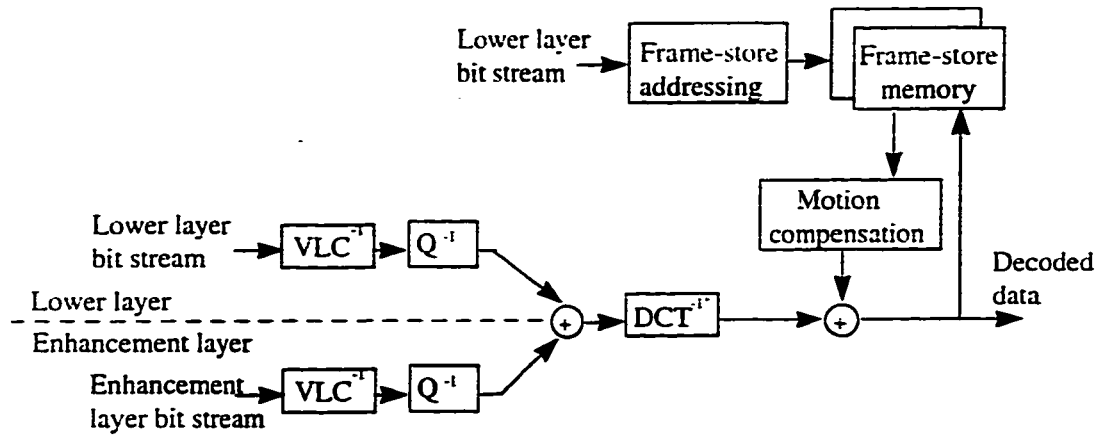


Fig. 3.5. Illustration of decoding process for SNR scalability

3.3.2 MPEG Spatial Scalability

Spatial scalability involves generating two spatial resolution video layers from a single video source, where the lower layer is coded by itself to provide the basic spatial resolution, and the enhancement layer is spatially interpolated from the lower layer and carries the full spatial resolution of the video source [4].

In MPEG spatial scalability, to code the higher (enhancement) layer, two kinds of predictions are required, which are obtained from the enhancement layer and lower layer, respectively (Fig. 3.6).

A motion compensated temporal prediction is performed from the previous decoded pictures in the enhancement layer. Meanwhile, a spatial prediction is formed from the up-sampled version of the lower layer decoded picture. This prediction is based on macroblocks. It is executed by up-sampling the corresponding region in the lower layer to the same grid of the current macroblock in the enhancement layer. For each block in a macroblock, the two predictions are selected or combined to form the final prediction. Hence, the macroblock type in the enhancement layer can be temporal prediction only, spatial prediction only, or weighted

combination of temporal and spatial predictions. In order to reconstruct the block pixels, the prediction is added to the prediction error, which is obtained by executing inverse DCT on the difference block DCT coefficients. A simplified decoding process for spatial scalability is illustrated in Fig. 3.7.

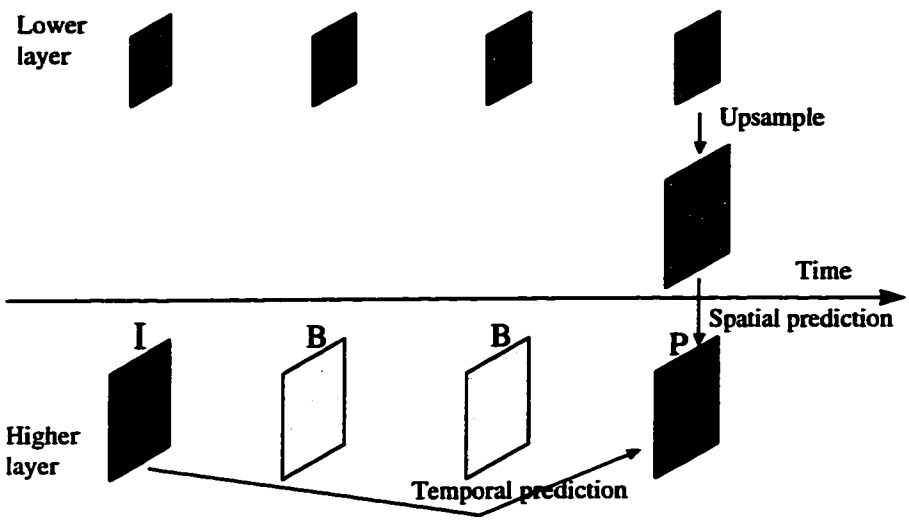


Fig. 3.6. Predictions for P-frames in spatial scalability coding

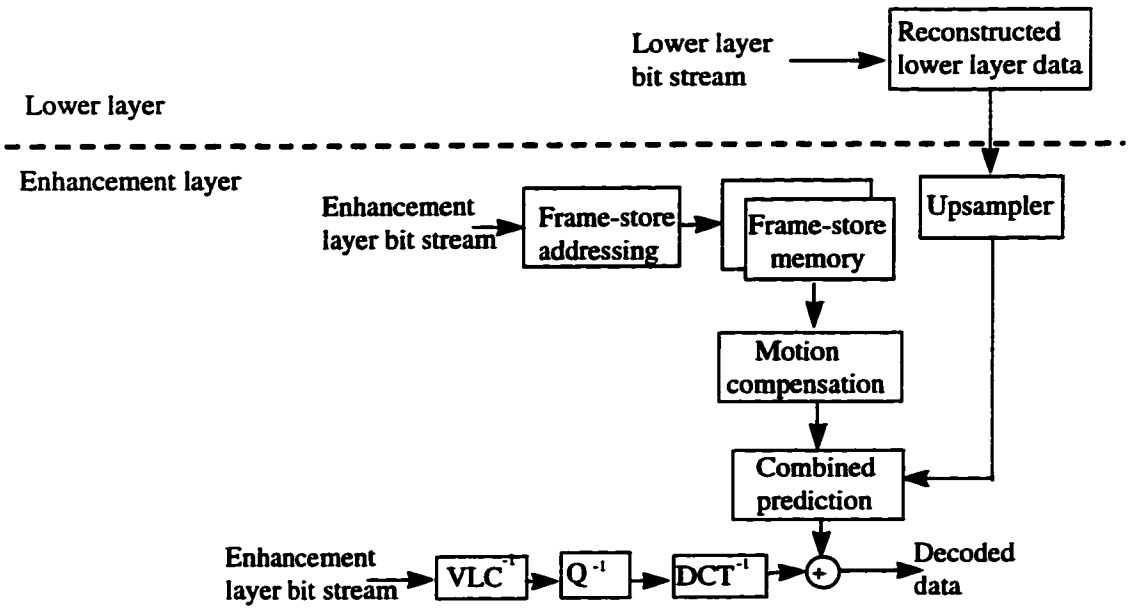


Fig. 3.7. Illustration of decoding process for spatial scalability

3.3.3 MPEG Temporal Scalability

Temporal scalability involves partitioning the video frames into layers, where the lower layer is coded by itself to provide the basic temporal rate, and the enhancement layer is temporal predicted with respect to the lower layer. These layers when temporal multiplexed yield full temporal resolution of the video source [4].

In MPEG temporal scalability, the decoding process for the enhancement layer is similar to the normal decoding process for a single layer (section 2.6). The only difference is that the prediction pictures for the enhancement layer can be selected from both the lower layer and the enhancement layer.

In P-pictures, the forward reference picture can be one of the following three: the most recent enhancement picture, the most recent lower layer picture, or the next lower layer picture in the display order (Fig. 3.8). In B-pictures, the forward reference can be either the most recent enhancement layer picture or the most recent lower layer picture, whereas the backward reference can be the most recent lower layer picture or the next lower layer picture in the display order (Fig. 3.9).

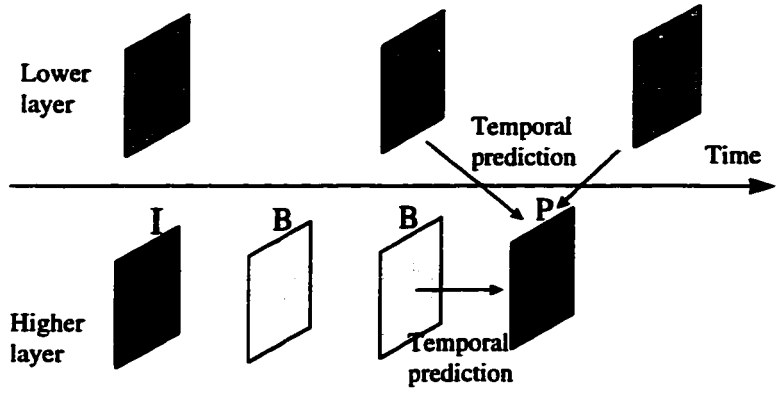


Fig. 3.8. Predictions for P-frames in temporal scalability coding

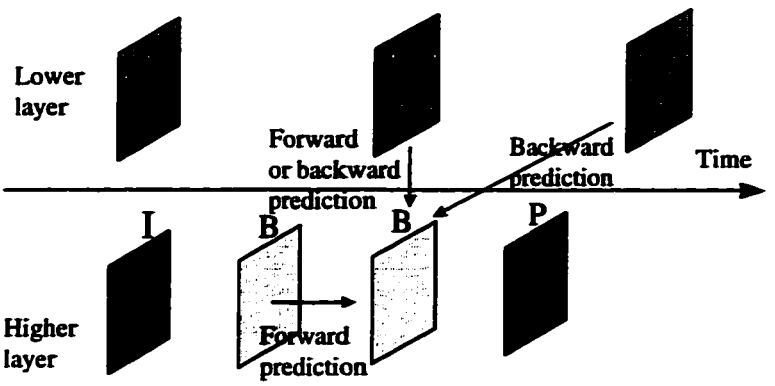


Fig. 3.9. Predictions for B-frames in temporal scalability coding

We note backward prediction cannot be made from a picture in the enhancement layer, which avoids the need for frame reordering in the enhancement layer. The enhancement layer can contain I-, P- or B-pictures. However, unlike the normal coding process, B-pictures can be used to predict the following P- or B-pictures in the enhancement layer. Hence, B-pictures in enhancement layer behave more like P-pictures.

We illustrate the decoding process for temporal scalability as follows.

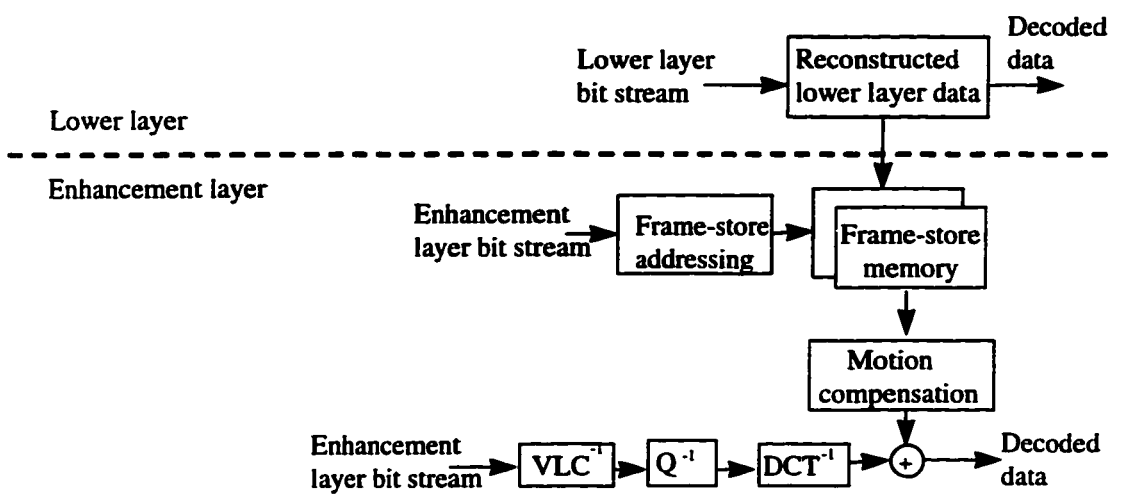


Fig. 3.10. Illustration of decoding process for temporal scalability

3.4 Summary

In this chapter, we have first introduced the basic concepts of scalable image/video compression including SNR scalability, spatial scalability and temporal scalability. This was followed by a review of spatial domain and transform domain coding techniques for SNR and spatial scalability. For SNR scalability, the spatial domain techniques are performed on the image pixel basis, such as bit plane and sub-sampling, while the transform domain techniques can be classified into coefficient scanning, bit slicing, and S-transform. For spatial scalability, the spatial domain techniques are usually based on pyramidal data structures. In addition, they are often used in conjunction with the transform domain techniques such as DCT and S-transform to provide a good coding performance.

We then presented the JPEG scalability coding schemes: progressive (SNR scalability) and hierarchical (spatial scalability) modes. Finally, the scalable features in the MPEG-2 standard, namely SNR scalability, spatial scalability and temporal scalability were discussed.

We note that the SNR scalability algorithms in both the JPEG and MPEG standards are implemented on the block DCT coefficients, which are generated by using the single layer compression scheme. Therefore, it is possible to update a single layer bit stream to a SNR scalability bit stream directly in the DCT compressed domain without fully decoding the bit stream back to the spatial domain. This operation basically only involves partitioning DCT coefficients into several bands or bit planes and reorganizing them into different coding layers.

However, the spatial scalability defined in the JPEG and MPEG standards relies on down-sampling and up-sampling an image/video frame in the spatial domain before DCT can be executed. To scale down a coded image/video sequence, or to update a single layer bit stream to a spatial scalable bit stream, a straight forward approach is to first fully decode the bit stream

back to the spatial domain, and then perform the desired manipulations. This technique is referred to as the spatial domain technique. In chapter 4 and chapter 5, we will propose a novel technique to implement the equivalent scaling and spatial scalability operations, which are fully performed in the DCT domain.

Chapter 4

Image/Video Spatial Scalability in Compressed Domain

In this chapter, we first address the need for manipulations of visual data in the compressed domain. Then we propose two DCT domain techniques to implement image/video spatial scalability. In section 4.2, the format compatible (FC) DCT technique is discussed. This technique is for manipulation of the standard bit streams, such as JPEG, MPEG, etc. Based on FC-DCT, the JPEG DCT domain technique for spatial scalability is presented, which is functionally equivalent to the standard JPEG spatial domain technique. Next, the format modified (FM) DCT technique is detailed in section 4.3. This technique is simple and computationally efficient. Finally, simulation results are provided in section 4.4, where the two DCT domain techniques are compared with the corresponding spatial domain techniques.

4.1 Need for Manipulations of Visual Data in Compressed Domain

Image/video compression is widely used in various applications with the advent of the JPEG, MPEG and H.261 standards. Hence most visual data are stored in the compressed format. Spatial scalable encoding of visual data has several applications including browsing visual databases, querying multimedia databases, interactive multimedia communications, etc.

There is an increasing interest in manipulating visual information directly in the compressed domain [15-21]. Smith and Rowe derived a family of algorithms to operate on the DCT domain compressed image data, such as algebraic operations (including pixel-wise and scalar addition and multiplication) and linear global operations [15, 16]. The algorithm for scaling operation is based on certain approximations. In addition, the detailed implementation is not provided, and the criteria for the approximations have to be refined. Chang and Messerschmitt derived many compositing operations for manipulating the compressed data. such as overlapping, scaling, translation and linear filtering, and the techniques to composite video directly in the DCT domain [17-20]. Natarajan and Vasudev have proposed a fast approximate algorithm for scaling down digital images in the DCT domain [21]. This algorithm has fewer operations compared to Chang and Messerschmitt's approach for scaling DCT coded images and video. However, the techniques proposed in the literature do not address the aspect of the implementation for image and video spatial scalability in the DCT domain as an extension of the respective scaling algorithms.

In the following sections, we will propose two novel techniques namely format compatible (FC) DCT and format modified (FM) DCT to implement image/video spatial scalability

directly on the DCT compressed data. The FC-DCT technique can be used to manipulate the standard bit streams, such as JPEG, MPEG, etc., and generate the compatible scaled image and video bit streams without changing the basic formats, i.e. the size of the DCT block. We note that in [15-21], the same format is used. However, the FC-DCT technique has a significantly higher computational complexity than the FM-DCT technique. The FM-DCT technique relies on the modification of the basic DCT block size for different levels of spatial scalability. This technique is simple and computationally efficient. Hence it can be employed in a variety of applications which require fast processing, such as indexing, retrieval and browsing through different resolutions of coded image/video. In contrast to the traditional spatial domain techniques, the compressed domain techniques remove the unnecessary decompression and re-compression procedures, thus having the advantages of reduced computational complexity and storage requirements.

4.2 FC-DCT Technique for Spatial Scalability

Recall from chapter 3 that the spatial scalability approach proposed in the JPEG and MPEG standards uses a spatial domain technique combined with a transform domain technique to achieve high compression. The spatial scalability encoder provides a pyramidal structure for encoding different resolutions of the image.

We now present a novel approach, i.e. the JPEG DCT domain technique to implement spatial scalability directly on the JPEG DCT coefficients within the framework of the JPEG standard. In this approach, the size of the DCT blocks is maintained to be 8x8 at all levels of scalability, which is identical to the JPEG and MPEG standards. From an implementation point of view, this approach is equivalent to the standard JPEG spatial domain technique. With some

modifications for motion compensation, a similar method can be extended to implement MPEG spatial scalability in the DCT domain.

To start with, we need to partially decode the JPEG bit stream to get access to the DCT coefficients, i.e. perform entropy decoding and de-quatization to derive the block DCT coefficients of the full size image. We then successively mean filter and down-sample the image coefficients in the block-based DCT domain to generate the block DCT coefficients corresponding to several lower resolution images. For example, when the image is scaled down by half, the basic DCT blocks in the original and scaled images are as shown in Fig. 4.1. It can be seen that a new block B in the preceding lower layer is related to four adjacent blocks B_1, B_2, B_3, B_4 in the current layer. The pixel values of block B can be calculated through equation (4.1), which is a linear operation. Since DCT is a unitary orthonormal transform, it is distributive to matrix multiplication. Therefore, the DCT of block B can be obtained through equation (4.2), i.e. the DCT coefficients of the block B can be generated from blocks B_1, B_2, B_3, B_4 directly in the DCT domain [18]. This procedure can be executed recursively to derive the DCT coefficients of the different layers.

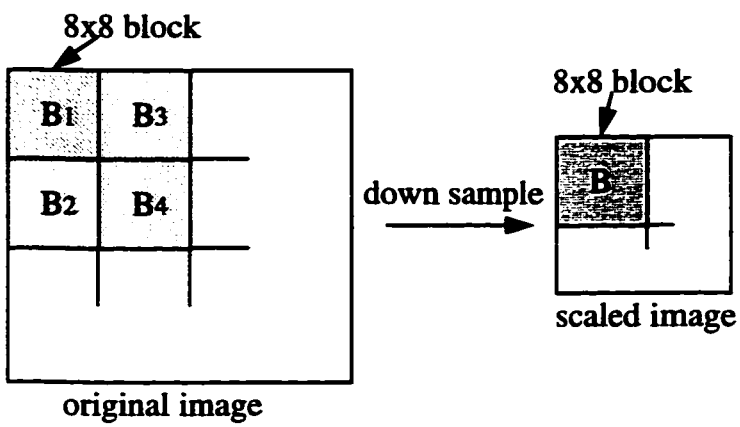


Fig. 4.1. Basic DCT blocks for FC-DCT domain technique

$$B = \sum_{i=1}^{4} H_i B_i W_i \tag{4.1}$$

$$DCT(B) = \sum_{i=1}^{4} DCT(H_i) DCT(B_i) DCT(W_i) \tag{4.2}$$

where

$$H_1 = H_3 = \begin{bmatrix} 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad H_2 = H_4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 \end{bmatrix}$$

$W_1 = W_2 = H_1^T$

$W_3 = W_4 = H_2^T$

$DCT(B_i)\ (i = 1, \cdots, 4)$ are available from the adjacent higher layer.

$DCT(H_i)$ and $DCT(W_i)$ are fixed, and can be calculated apriori.

The proposed FC-DCT technique for JPEG spatial scalable encoding would execute all the operations in the DCT domain based on the DCT coefficients of the down-sampled images for different layers. We call this encoder as the JPEG DCT domain hierarchical mode encoder. There are two major differences between this encoder and the standard JPEG hierarchical mode encoder. First, the JPEG DCT domain encoder removes decompression and re-compression operations, i.e. inverse DCT and forward DCT. Second, it involves up-sampling operations in the DCT domain instead of pixel domain. The DCT domain up-sampling operation can be derived in a similar manner as the down-sampling operation. A block of the lower layer A is interpolated and up-sampled by a factor of two to form four blocks A_1, A_2, A_3, A_4 of the

preceding higher layer by using equation (4.3) (Fig. 4.2). Due to the linearity of the up-sampling function and the distributive property of DCT, the DCT coefficients of blocks A_1, A_2, A_3, A_4 can be calculated using equation (4.4).

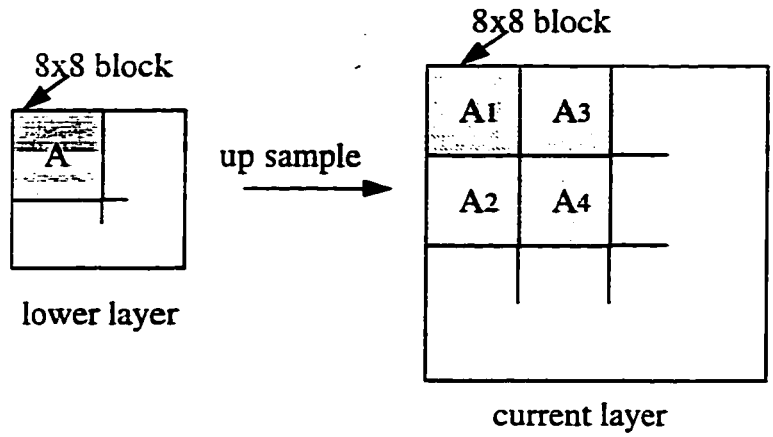


Fig. 4.2. Up-sampling operation for a block in pixel domain

$$
\begin{aligned}
A_1 &= UAU^T \\
A_2 &= UAV^T \\
A_3 &= VAU^T \\
A_4 &= VAV^T
\end{aligned}
\tag{4.3}
$$

$$
\begin{aligned}
DCT(A_1) &= DCT(U)DCT(A)DCT(U^T) \\
DCT(A_2) &= DCT(U)DCT(A)DCT(V^T) \\
DCT(A_3) &= DCT(V)DCT(A)DCT(U^T) \\
DCT(A_4) &= DCT(V)DCT(A)DCT(V^T)
\end{aligned}
\tag{4.4}
$$

where

$$
U = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0
\end{bmatrix}
\qquad
V = \begin{bmatrix}
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
$$

$DCT(A)$ is available from the adjacent lower layer.

$DCT(U)$, $DCT(U^T)$, $DCT(V)$ and $DCT(V^T)$ are fixed, and can be calculated apriori.

The procedure of the JPEG DCT domain hierarchical mode encoder is illustrated in Fig. 4.3. To encode the lowest layer I_0, only quantization operation on the DCT coefficients is required, since the data is already available in the form of DCT coefficients for each 8x8 block of the image. To encode the next higher layer I_1, I_0 is interpolated and up-sampled in the DCT domain to form U_0 as the prediction for I_1. The difference DCT data (D_1) between I_1 and U_0 is thus available immediately which is then quantized. The DCT data R_1 for the reconstructed image can be obtained by adding D_1 to U_0. This procedure is recursively executed to encode other higher layers until the full resolution layer.

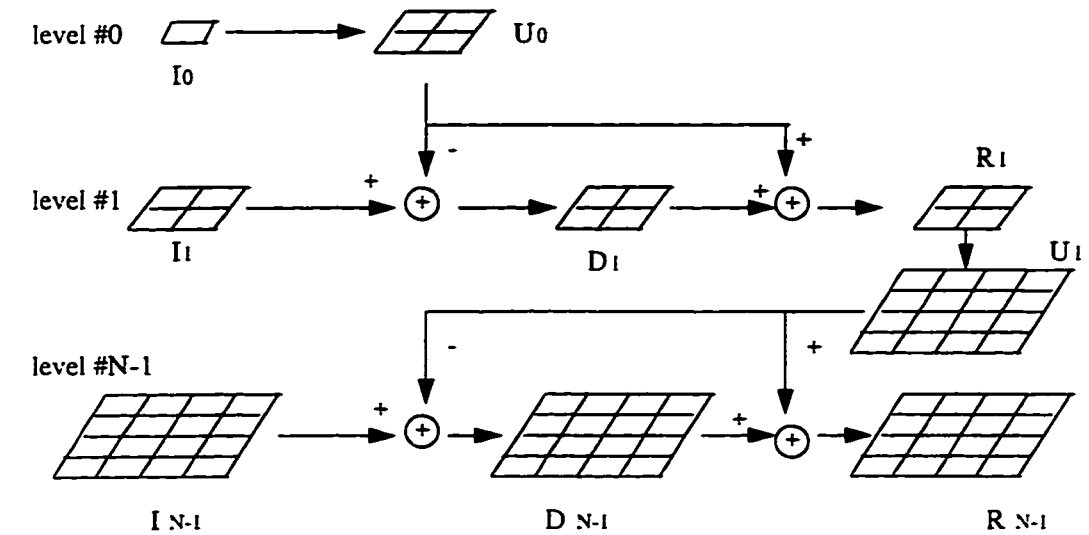


Fig. 4.3. JPEG DCT domain hierarchical mode encoder

It can be seen that the major computations of the proposed JPEG DCT domain technique is to perform down-sampling and up-sampling operations on the image coefficients in the DCT domain, which essentially involve only pre-matrix and post-matrix multiplications. Note that the pre-matrix and post-matrix are sparse matrices. Furthermore, there are many zero DCT coefficients in the transform blocks of the image after quantization. We can therefore exploit these two features to further reduce the computational complexity.

4.3 FM-DCT Technique for Spatial Scalability

We recall from section 3.1.2, that the transform domain techniques for image spatial scalability can be based on either frequency or spatial pyramids. In the JPEG and MPEG standards, the spatial pyramid is used, in which the size of the DCT block is the same at all levels of scalability. In section 4.2, we have presented the FC-DCT technique based on this spatial pyramid.

We note that the energy of most natural images is concentrated in the lower order coefficients. As a result, smaller spatial resolution images typically contain lower spatial frequencies of the original image. Therefore, it is possible to reconstruct an image in a hierarchical format by reorganizing the transform coefficients in a pyramidal fashion. In this section, we will propose a novel approach called FM-DCT to implement spatial scalability in the DCT compressed domain, which is based on the frequency pyramid. The DCT block size is scaled corresponding to the level. For example, if a 512x512 image has 8x8 DCT blocks, then a 256x256 image would have 4x4 DCT blocks. The most attractive features of this approach are its simplicity and efficiency. In addition, with the smaller size of the DCT blocks in the scaled

image, the complexity of the decoder is also reduced if decoding of the scaled image is required, which is a clear advantage for software decoding.

4.3.1 Relationship between Original and Scaled Images

To implement spatial scalability in the DCT domain, the DCT coefficients of several lower resolution images are derived directly from the DCT coefficients of the original image. Suppose the basic DCT block B in the original image is of size NxN, which is represented by the matrix $f(i,j)$ $(i,j=0,1,\cdots,N-1)$. In the spatial domain, down-sampling is a trivial operation. In a simple case, the DCT block B' in the scaled image $f'(i,j)$ can be generated by down sampling $f(i,j)$ with a scalar m described as follows:

$$f'(i,j) = f(m\cdot i, m\cdot j) \quad \text{for } i,j = 0,1,\cdots,(\tfrac{N}{m})-1 \tag{4.5}$$

where $m = 2^p$, and p is an integer.

Since the DCT is a global operation, and each block DCT coefficient is based on all the pixels in the block of the image, a similar relationship between the original and scaled blocks is not directly applicable in the transform domain.

The forward DCT on the data $f(i,j)$ is defined as follows:

$$F(u,v) = \frac{2}{N}C(u)C(v)\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} f(i,j)\cos\frac{(2i+1)u\pi}{2N}\cos\frac{(2j+1)v\pi}{2N} \tag{4.6}$$

for $u,v = 0,1,\cdots,N-1$

where $C(x) = \begin{cases} \frac{1}{\sqrt{2}} & x=0 \\ 1 & \text{otherwise} \end{cases}$

Similarly, the forward DCT on the data $f'(i,j)$ can be written as:

$$F'(u,v) = \frac{2}{\frac{N}{m}} C(u)C(v) \sum_{i=0}^{(\frac{N}{m})-1} \sum_{j=0}^{(\frac{N}{m})-1} f'(i,j) \cos\frac{(2i+1)u\pi}{2\cdot\frac{N}{m}} \cos\frac{(2j+1)v\pi}{2\cdot\frac{N}{m}}$$

$$= \frac{2}{\frac{N}{2^p}} C(u)C(v) \sum_{i=0}^{(\frac{N}{2^p})-1} \sum_{j=0}^{(\frac{N}{2^p})-1} f(2^p\cdot i, 2^p\cdot j) \cos\frac{(2i+1)u\pi}{2\cdot\frac{N}{2^p}} \cos\frac{(2j+1)v\pi}{2\cdot\frac{N}{2^p}} \qquad (4.7)$$

$$\text{for } u,v = 0,1,\cdots,(\frac{N}{2^p})-1$$

From equations (4.6) and (4.7), it can be seen that the frequencies of the cos(·) terms in the cosine transform matrices are dependent on the size of the DCT block, hence the frequencies of the original and scaled blocks are quite different. In order to establish a relationship between equations (4.6) and (4.7), it is necessary to modify the frequencies of the cos(·) terms. We note that this is a computationally intensive operation. Since our intent is to efficiently calculate $F(u,v)$ from $F'(u,v)$,we are interested in deriving a simple relationship between equations (4.6) and (4.7).

4.3.2 Spatial Scalability in DCT Domain

We will now show that under certain assumptions, a simple relationship between the original and scaled blocks in the image can be obtained in the DCT domain. In other words, the DCT coefficients of the scaled block can be efficiently generated from the DCT coefficients of the original block. For the sake of simplicity, we first discuss the one dimensional case, and will later extend this to two dimensions.

One dimensional operation

In the one dimensional case, the down-sampled sequence $f'(i)$ can be generated by selecting one point from every m points in the original sequence $f(i)$ as follows:

$$f'(i) = f(m \cdot i) \qquad \text{for } i = 0,1,\cdots,(\frac{N}{m}) - 1 \tag{4.8}$$

The forward DCT of the original and sampled 1-D sequences are similar to equations (4.6) and (4.7), and defined as follows:

$$F(u) = \sqrt{\frac{2}{N}} C(u) \sum_{i=0}^{N-1} f(i) \cos \frac{(2i+1)u\pi}{2N} \qquad \text{for } u = 0,1,\cdots,N-1 \tag{4.9}$$

$$F'(u) = \sqrt{\frac{2}{\frac{N}{2^p}}} C(u) \sum_{i=0}^{(\frac{N}{2^p})-1} f(2^p \cdot i) \cos \frac{(2i+1)u\pi}{2 \cdot \frac{N}{2^p}} \qquad \text{for } u = 0,1,\cdots,(\frac{N}{2^p}) - 1 \tag{4.10}$$

To approximate $F'(u)$ to $F(u)$, we rewrite equation (4.10) as follows:

$$F(u) = \sqrt{\frac{2}{N}} C(u) \sum_{i=0}^{(\frac{N}{2^p})-1} \left[\begin{array}{l} f(2^p \cdot i) \cos \dfrac{(2 \cdot 2^p \cdot i + 1)u\pi}{2N} \\ + f(2^p \cdot i + 1) \cos \dfrac{(2 \cdot 2^p \cdot i + 3)u\pi}{2N} \\ + f(2^p \cdot i + 2) \cos \dfrac{(2 \cdot 2^p \cdot i + 5)u\pi}{2N} \\ + \\ \vdots \\ + f(2^p \cdot i + 2^p - 1) \dfrac{(2 \cdot 2^p \cdot i + 2^{p+1} - 1)u\pi}{2N} \end{array} \right] \tag{4.11}$$

When $m = 2^p$ is small, we can assume that

$$f(2^p \cdot i) = f(2^p \cdot i + 1) = f(2^p \cdot i + 2) = \cdots = f(2^p \cdot i + 2^p - 1) \tag{4.12}$$

Therefore, equation (4.11) can be modified as:

$$F(u)=\sqrt{\frac{2}{N}}C(u)\sum_{i=0}^{(\frac{N}{2^p})-1} f(2^p\cdot i)\cdot\left[\begin{array}{l}\cos\dfrac{(2\cdot 2^p\cdot i+1)u\pi}{2N}\\ +\cos\dfrac{(2\cdot 2^p\cdot i+3)u\pi}{2N}\\ +\cos\dfrac{(2\cdot 2^p\cdot i+5)u\pi}{2N}\\ +\\ \vdots\\ +\cos\dfrac{(2\cdot 2^p\cdot i+2^{p+1}-1)u\pi}{2N}\end{array}\right] \tag{4.13}$$

It can be observed that there are 2^p summation terms in equation (4.13). After combining the adjacent terms, we can further modify equation (4.13) as:

$$F(u)=\sqrt{\frac{2}{N}}C(u)\sum_{i=0}^{(\frac{N}{2^p})-1} f(2^p\cdot i)\cdot 2\cos\frac{u\pi}{\frac{4N}{2}}\cdot\left[\begin{array}{l}\cos\dfrac{(2^p\cdot i+1)u\pi}{N}\\ +\cos\dfrac{(2^p\cdot i+3)u\pi}{N}\\ +\cos\dfrac{(2^p\cdot i+5)u\pi}{N}\\ +\\ \vdots\\ +\cos\dfrac{(2^p\cdot i+2^p-1)u\pi}{N}\end{array}\right] \tag{4.14}$$

To further simplify this, we recursively combine the adjacent terms in equation (4.14) as follows:

$$F(u) = \sqrt{\frac{2}{N}}C(u)\sum_{i=0}^{(\frac{N}{2^p})-1} f(2^p \cdot i)\cdot 2^2 \cdot \cos\frac{u\pi}{\frac{4N}{2}}\cdot\cos\frac{u\pi}{\frac{4N}{2^2}}\cdot \begin{bmatrix} \cos\dfrac{(2^{p-1}\cdot i+1)u\pi}{\frac{N}{2}} \\ +\cos\dfrac{(2^{p-1}\cdot i+3)u\pi}{\frac{N}{2}} \\ + \\ \vdots \\ +\cos\dfrac{(2^{p-1}\cdot i+2^{p-1}-1)u\pi}{\frac{N}{2}} \end{bmatrix}$$

$$\vdots$$

$$= \sqrt{\frac{2}{N}}C(u)\sum_{i=0}^{(\frac{N}{2^p})-1} f(2^p \cdot i)\cdot 2^{p-1}\cdot\cos\frac{u\pi}{\frac{4N}{2}}\cdots\cos\frac{u\pi}{\frac{4N}{2^{p-1}}}\cdot\left[\cos\frac{(4i+1)u\pi}{\frac{N}{2^{p-2}}}+\cos\frac{(4i+3)u\pi}{\frac{N}{2^{p-2}}}\right]$$

$$= \sqrt{\frac{2}{N}}C(u)\sum_{i=0}^{(\frac{N}{2^p})-1} f(2^p \cdot i)\cdot 2^p\cdot\cos\frac{u\pi}{\frac{4N}{2}}\cdot\cos\frac{u\pi}{\frac{4N}{2^2}}\cdots\cos\frac{u\pi}{\frac{4N}{2^p}}\cdot\cos\frac{(2i+1)u\pi}{\frac{N}{2^{p-1}}}$$

$$= 2^{p/2}\cdot\cos\frac{u\pi}{\frac{4N}{2}}\cdot\cos\frac{u\pi}{\frac{4N}{2^2}}\cdots\cos\frac{u\pi}{\frac{4N}{2^p}}\cdot F'(u) \qquad (4.15)$$

From equation (4.15), a simple relationship between the coefficients $F'(u)$ of the sampled sequence and $F(u)$ of the original sequence can be obtained as follows:

$$F'(u) = \frac{F(u)}{2^{p/2}\cdot\cos\frac{u\pi}{\frac{4N}{2}}\cdot\cos\frac{u\pi}{\frac{4N}{2^2}}\cdots\cos\frac{u\pi}{\frac{4N}{2^p}}} \qquad \text{for } u = 0,1,\cdots,(\frac{N}{2^p})-1 \qquad (4.16)$$

It can be seen that the DCT coefficients of the sampled sequence $F'(u)$ are directly obtained from the corresponding DCT coefficients of the original sequence $F(u)$ with only a scale factor. Hence, this is a computational efficient operation. In addition, we can pre-calculate and store the scale factors, which further improves the speed of computation. We note that

$F'(u)$ is an approximate result, since we rely on the assumptions stated in equation (4.12). However, when $m=2^p$ is small, the assumptions are reasonable, especially for highly correlated data.

Two dimensional operation

The proposed approach to derive the DCT coefficients of the down-sampled sequence can be directly extended to two dimensions. Note that equation (4.6) can be rewritten in the following form:

$$F(u,v)=\frac{2}{N}C(u)C(v)\sum_{i=0}^{(\frac{N}{2^p})-1}\sum_{j=0}^{(\frac{N}{2^p})-1}\left[\begin{array}{l}
f(2^p\cdot i,2^p\cdot j)\cos\frac{(2\cdot 2^p\cdot i+1)u\pi}{2N}\cos\frac{(2\cdot 2^p\cdot j+1)v\pi}{2N}\\
+f(2^p\cdot i+1,2^p\cdot j)\cos\frac{(2\cdot 2^p\cdot i+3)u\pi}{2N}\cos\frac{(2\cdot 2^p\cdot j+1)v\pi}{2N}\\
+\\
\vdots\\
+f(2^p\cdot i+2^p-1,2^p\cdot j)\cos\frac{(2\cdot 2^p\cdot i+2^{p+1}-1)u\pi}{2N}\cos\frac{(2\cdot 2^p\cdot j+1)v\pi}{2N}\\
+f(2^p\cdot i,2^p\cdot j+1)\cos\frac{(2\cdot 2^p\cdot i+1)u\pi}{2N}\cos\frac{(2\cdot 2^p\cdot j+3)v\pi}{2N}\\
+f(2^p\cdot i+1,2^p\cdot j+1)\cos\frac{(2\cdot 2^p\cdot i+3)u\pi}{2N}\cos\frac{(2\cdot 2^p\cdot j+3)v\pi}{2N}\\
+\\
\vdots\\
+f(2^p\cdot i+2^p-1,2^p\cdot j+1)\cos\frac{(2\cdot 2^p\cdot i+2^{p+1}-1)u\pi}{2N}\cos\frac{(2\cdot 2^p\cdot j+3)v\pi}{2N}\\
+\\
\vdots\\
+f(2^p\cdot i+2^p-1,2^p\cdot j+2^p-1)\cos\frac{(2\cdot 2^p\cdot i+2^{p+1}-1)u\pi}{2N}\cos\frac{(2\cdot 2^p\cdot j+2^{p+1}-1)v\pi}{2N}
\end{array}\right] \quad (4.17)$$

When $m = 2^p$ is small, we can assume that the $f(i,j)$ values within each sampling square are approximately equal. Similar to the one dimensional case, we can modify equation (4.17) to obtain the following equation:

$$F'(u,v) = \frac{F(u,v)}{2^p \cdot \cos\frac{u\pi}{\frac{4N}{2}} \cdot \cos\frac{v\pi}{\frac{4N}{2}} \cdot \cos\frac{u\pi}{\frac{4N}{2^2}} \cdot \cos\frac{v\pi}{\frac{4N}{2^2}} \cdots \cos\frac{u\pi}{\frac{4N}{2^p}} \cos\frac{v\pi}{\frac{4N}{2^p}}} \tag{4.18}$$

for $u, v = 0,1,\cdots,(\frac{N}{2^p}) - 1$

Hence, the DCT coefficients of the scaled block can be directly obtained from the corresponding coefficients of the original block with only a scale factor. The proposed method provides a simple and efficient way of implementing the down-sampling operation in the DCT domain. For example, when the image is scaled down by half, the DCT block in the scaled image will be reduced to $\frac{N}{2} x \frac{N}{2}$ (Fig. 4.4).

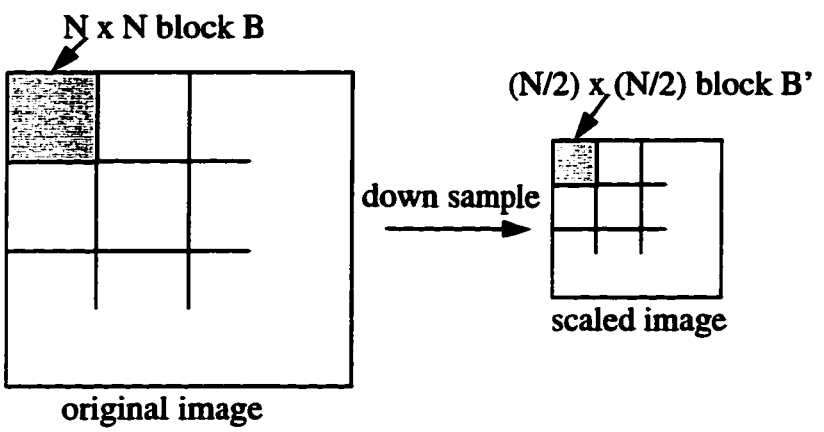


Fig. 4.4. Basic DCT blocks for FM-DCT domain technique

The relationship of the DCT coefficients between the corresponding blocks B and B' in the original and scaled images is shown in equation (4.19). The proposed technique for scaling an image in the DCT domain is also flexible and scalable. We can obtain the desired size block by using equation (4.19) recursively to progressively scale down the image by $\frac{1}{2}, \frac{1}{4}, \cdots$, etc.

$$F'(u,v) = \frac{F(u,v)}{2\cos\frac{u\pi}{2N}\cos\frac{v\pi}{2N}} \qquad \text{for } u,v = 0,1,\cdots,(\frac{N}{2})-1 \tag{4.19}$$

We note that the value of m depends on the size the DCT block, and its range is between 1 and N. With the increase of m, the quality of the scaled image decreases. In practice, m can be chosen based on the quality requirement.

4.4 Simulation Results

Simulations have been performed using the CCITT standard test image "Lena" of size 512x512 pixels quantized to 8 bits per pixel. We first apply the JPEG baseline sequential scheme to encode the "Lena" image without scalability. Then, the FM-DCT and FC-DCT techniques for spatial scalability are executed on the JPEG DCT coefficients. In this thesis, we have chosen three layers of spatial scalability, i.e. the images are represented in sizes of 512x512, 256x256 and 128x128, respectively. The relative performance of the two techniques are evaluated and compared with the spatial domain techniques from the perspectives of computational complexity and image quality. For the spatial domain technique, we use a fast algorithm [22] to evaluate the computational complexity of the forward/inverse DCT, which has a complexity of $2\log_2 N - 3 + \frac{8}{N}$ multiplications and $3(\log_2 N - 1) + \frac{4}{N}$ additions per pixel.

The objective quality is evaluated using peak signal to noise ratio (PSNR), which is defined in equation (4.20). All the reconstructed scaled images are up-sampled to the full size of 512x512 pixels, therefore the PSNR for each image can be calculated with respect to the original 512x512 image.

$$PSNR = 10\log_{10}\left(\frac{(Peak\ signal\ value)^2}{Mean\ square\ error}\right) \tag{4.20}$$

where $Mean\ square\ error = \frac{1}{N^2}\sum_{i,j}\left(x_{ij} - y_{ij}\right)^2$

Peak signal value = 255 for an 8 bits/pixel image

x_{ij}, y_{ij} = value of pixel (i, j) in the original and reconstructed images, respectively

N^2 = number of pixels in the image

4.4.1 FC-DCT Technique

The FC-DCT and the spatial domain techniques are executed to generate two JPEG spatial scalable bit streams for comparison. For the FC-DCT technique, we take advantage of the sparse matrix, and execute computations corresponding to only the non-zero coefficients.

The complexities of the down-sampling operations and hierarchical encoding on the compressed "Lena" image for the FC-DCT and spatial domain approaches are tabulated in Table 4.1 and Table 4.2.1-4.2.3, respectively. The total number of operations is therefore calculated by summation of the individual column in Table 4.1 and Table 4.2.1-4.2.3 weighted by the number of pixels. Since the total number of operations depends on the level of scalability, we have chosen to represent the computational complexity in terms of number of multiplications and additions per pixel.

Table 4.1. Computational complexity of down-sampling operations for DCT domain vs. spatial domain techniques

Operations	DCT domain		Spatial domain	
	mul # /pixel	add # /pixel	mul # /pixel	add # /pixel
IDCT (512x512)	none	none	4 (512x512)	6.5 (512x512)
Down-sample (512x512 to 256x256)	1.44 (512x512)	2.19 (512x512)	0.25 (512x512)	0.75 (512x512)
Down-sample (256x256 to 128x128)	5.56 (256x256)	6.31 (256x256)	0.25 (256x256)	0.75 (256x256)

Table 4.2.1. Computational complexity of JPEG spatial scalable encoding for FC-DCT vs. spatial domain techniques (128x128 layer)

Operations	DCT domain		Spatial domain	
	mul # /pixel	add # /pixel	mul # /pixel	add # /pixel
FDCT (128x128)	none	none	4 (128x128)	6.5 (128x128)

Table 4.2.2. Computational complexity of JPEG spatial scalable encoding for FC-DCT vs. spatial domain techniques (256x256 layer)

Operations	DCT domain		Spatial domain	
	mul # /pixel	add # /pixel	mul # /pixel	add # /pixel
IDCT (128x128)	none	none	4 (128x128)	6.5 (128x128)
Up-sample (128x128 to 256x256)	2.48 (256x256)	2.48 (256x256)	0.75 (256x256)	1.25 (256x256)
FDCT (256x256)	none	none	4 (256x256)	6.5 (256x256)

Table 4.2.3. Computational complexity of JPEG spatial scalable encoding for FC-DCT vs. spatial domain techniques (512x512 layer)

Operations	DCT domain		Spatial domain	
	mul # /pixel	add # /pixel	mul # /pixel	add # /pixel
IDCT (256x256)	none	none	4 (256x256)	6.5 (256x256)
Up-sample (256x256 to 512x512)	3.62 (512x512)	3.62 (512x512)	0.75 (512x512)	1.25 (512x512)
FDCT (512x512)	none	none	4 (512x512)	6.5 (512x512)

From the calculations, the overall multiplications and additions per pixel on the 512x512 image are 7.07 and 8.01, respectively, for the FC-DCT approach, and 11.75 and 19.56, respectively, for the spatial domain approach. In the simulations, we have set the coefficients in the pre-matrix and post-matrix to zero when the absolute values are less than 0.01. Using this approximation, the DCT domain approach can reduce the number of multiplications and additions by 40% and 59%, respectively, compared to the spatial domain approach.

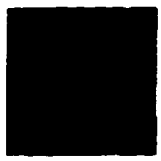
Although the FC-DCT technique performs all the operations fully in the DCT domain, it is functionally equivalent to the spatial domain technique. Therefore, the coded bit stream is compatible with the standard JPEG spatial scalable bit stream, which is a significant advantage of the proposed technique. The JPEG hierarchical mode decoder with minor modifications can be used to decode the FC-DCT coded bit stream and reconstruct the images of different layers. The coding performance of the DCT and spatial domain techniques is tabulated in Table 4.3. It can be seen that the FC-DCT technique achieves a comparable performance with the spatial domain technique.

Table 4.3. Performance comparison of FC-DCT vs. spatial domain techniques for spatial scalability

Image size	**PSNR (DCT domain) dB**	**PSNR (Spatial domain) dB**
128x128	24.61	24.61
256x256	28.20	28.22
512x512	33.50	33.44

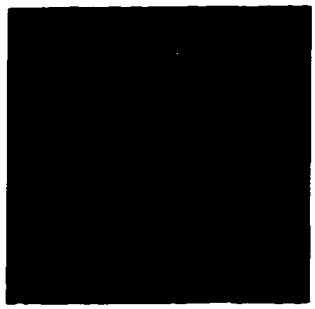
For the spatial domain and DCT domain techniques, the reconstructed "Lena" images of three layers are shown in Fig. 4.5, in which the image sizes are enlarged progressively. It can be observed that the subjective qualities of the reconstructed "Lena" images are almost identical for the two techniques. As more coefficients in the pre-matrix and post-matrix are set to zero, it

results in a reduced complexity, but at the expense of decrease in the quality of the reconstructed image. Hence, there is a tradeoff between the computational complexity and image quality.

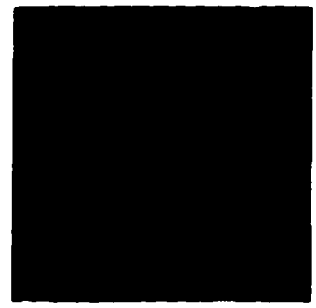


(a.1) Image size: 128x128

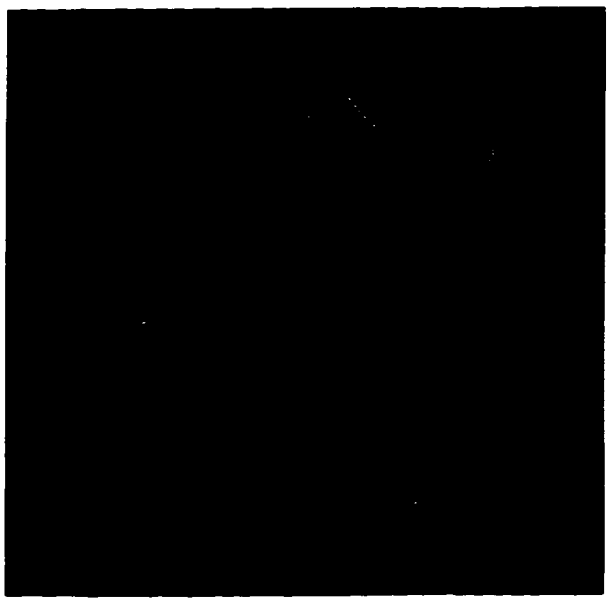
(b.1) Image size: 128x128



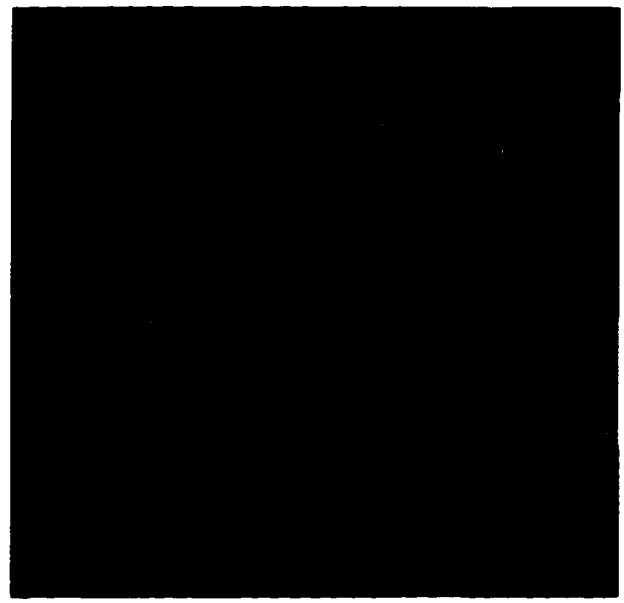
(a.2) Image size: 256x256



(b.2) Image size: 256x256



(a.3) Image size: 512x512



(b.3) Image size: 512x512

Fig. 4.5. Reconstructed three layer "Lena" images in JPEG hierarchical encoding
(a) Spatial domain technique (b) DCT domain technique

4.4.2 FM-DCT Technique

In the simulations, the FM-DCT and spatial domain techniques are executed to down-sample the image in the DCT and spatial domain, respectively. The basic DCT block sizes for the 512x512, 256x256 and 128x128 pixel images are 8x8, 4x4 and 2x2, respectively.

In the first experiment, we obtain the DCT coefficients of the down-sampled 256x256 and 128x128 images directly from the DCT coefficients of the original 512x512 image using the proposed technique. Recall from section 4.3 that the pixel values in each sampling square are assumed to be equal, which greatly reduces the computational complexity. To obtain each DCT coefficient of the scaled image, only one multiplication operation is required.

In the second experiment, the inverse DCT is first executed to reconstruct the 512x512 image. The computations for inverse DCT are tabulated in Table 4.4 (Note that the FM-DCT technique doesn't require these inverse DCT operations). Next, the image is down-sampled in the spatial domain to obtain the desired size. Finally, the forward DCT is executed on the smaller size images to maintain the data format in the DCT domain.

Table 4.4. Computational complexity of inverse DCT for spatial domain technique (512x512 layer, DCT block size: 8x8)

Operations	mul # /pixel	add # /pixel
IDCT (512x512)	4 (512x512)	6.5 (512x512)

The operations for the FM-DCT and spatial domain techniques for spatial scalable encoding of the different layer images are tabulated in Table 4.5.1 and 4.5.2. It is clear that the proposed FM-DCT technique is computationally efficient.

To compare the image qualities of these two techniques, we perform the inverse DCT on the coefficients generated by these two techniques to reconstruct the scaled images. The PSNR for the reconstructed "Lena" images is tabulated in Table 4.6. It can be seen that, compared to the spatial domain technique, the proposed FM-DCT technique introduces a marginal loss on quality which is image content dependent.

Table 4.5.1. Computational complexity of spatial scalable encoding for FM-DCT vs. spatial domain techniques (256x256 layer, DCT block size: 4x4)

Operations	DCT domain		Spatial domain	
	mul # /pixel	add # /pixel	mul # /pixel	add # /pixel
Down-sample (512x512 to 256x256)	0.25 (512x512)	none	0.25 (512x512)	0.75 (512x512)
FDCT (256x256)	none	none	3 (256x256)	4 (256x256)

Table 4.5.2. Computational complexity of spatial scalable encoding for FM-DCT vs. spatial domain techniques (128x128 layer, DCT block size: 2x2)

Operations	DCT domain		Spatial domain	
	mul # /pixel	add # /pixel	mul # /pixel	add # /pixel
Down-sample (256x256 to 128x128)	0.25 (256x256)	none	0.25 (256x256)	0.75 (256x256)
FDCT (128x128)	none	none	1 (128x128)	2 (128x128)

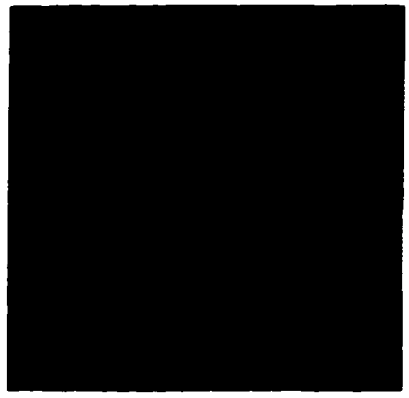
Table 4.6. PSNR of the scaled "Lena" images for FM-DCT vs. spatial domain techniques

Image size	PSNR (DCT domain) dB	PSNR (Spatial domain) dB
128x128	25.45	25.74
256x256	29.24	29.88

The reconstructed "Lena" images of two layers (256x256 and 128x128) for these two approaches are shown in Fig. 4.6. It can be observed that the subjective quality degradation of the reconstructed scaled "Lena" images for the DCT domain technique is almost unnoticeable.

(a.1) Image size: 128x128

(b.1) Image size: 128x128



(a.2) Image size: 256x256

(b.2) Image size:256x256

Fig. 4.6. Reconstructed scaled "Lena" image
(a) Spatial domain technique (b) DCT domain technique

To test the validity of the FC-DCT technique for different kinds of images, we have executed the same experiments on other two standard images: "Barbara" and "Baboon". Similar results have been obtained. The PSNR of the scaled "Barbara" and "Baboon" images are tabulated in Table 4.7 and 4.8, respectively. These results confirm that the proposed technique achieves a comparable performance at a substantial reduced computational complexity compared to the spatial domain technique.

Table 4.7. PSNR of the scaled "Barbara" images for FM-DCT vs. spatial domain techniques

Image size	PSNR (DCT domain) dB	PSNR (Spatial domain) dB
128x128	22.54	22.71
256x256	24.61	25.22

Table 4.8. PSNR of the scaled "Baboon" images for FM-DCT vs. spatial domain techniques

Image size	PSNR (DCT domain) dB	PSNR (Spatial domain) dB
128x128	20.70	20.84
256x256	22.60	23.19

4.5 Summary

In this chapter, the need for manipulations of visual data in the compressed domain is first reviewed. The research area of manipulating compressed visual data has emerged as a focus of increasing interest. However, the techniques proposed in the literature do not address the aspect of the implementation for visual spatial scalability in the DCT domain as an extension of the respective scaling algorithm. Therefore, in this thesis, we emphasize on the spatial scalability issue.

The FC-DCT technique for spatial scalability is first presented. We have shown that a spatial scalable JPEG bit stream can be generated from the baseline JPEG bit stream by operating directly on the JPEG DCT coefficients. From an implementation point of view, this approach is equivalent to the standard JPEG spatial domain technique. A JPEG hierarchical mode decoder with minor modifications can be used to decode the bit stream, where the reconstructed image is zoomed progressively through several passes.

Next, the FM-DCT technique for spatial scalability is detailed. The computational complexity is greatly reduced by using certain approximations. Therefore, this technique is simple and computationally efficient. It is suitable for applications which require fast processing.

The FC-DCT technique is targeted to manipulations of the standard bit streams. However, it is not directly applicable to video data, since motion estimation/compensation is involved for temporal redundancy removal. In chapter 5, we will propose an approach to implement encoding of the scaled MPEG video using a combined FC-DCT and modified DCT domain motion compensation technique.

Chapter 5

Encoding Scaled MPEG Video in Compressed Domain

In this chapter, we first introduce the problem of manipulation of MPEG video bit streams. It is shown that the FC-DCT technique is not directly applicable to MPEG compressed video due to motion estimation/compensation in the temporal dimension. In section 5.2, we propose a macroblock type selective (MTS) encoder to implement encoding scalable MPEG video directly in the motion-compensated DCT domain. We note that in order to encode the predictive macroblocks, the new motion vectors and DCT coefficients corresponding to the prediction error blocks can be obtained from the existing motion vectors and DCT coefficients. The macroblock type selection strategy (of the scaled macroblocks) in the proposed MTS-encoder and the experimental analysis are detailed in section 5.3. Finally, the summary is presented in section 5.4.

5.1 Introduction to Manipulations of MPEG Video

In advanced video services, further manipulations of the compressed visual information may be required. Scaling visual data in the compressed domain has many applications. For example, a simultaneous display of four video sources in a video conferencing system requires down scaling of each video source by half, and generating a composite video sequence. Another example is the need of spatial scalable video encoding for a multi-user environment, where a video sequence is transmitted and displayed at different resolutions based on the viewer's request. In addition, spatial scalability is valuable for video transmission over ATM networks, where the best video quality is to be achieved within resource limitations.

We note that a MPEG coded bit stream consists of both DCT data and motion information. Hence the FC-DCT technique proposed in chapter 4 is not directly applicable to the MPEG video due to the additional operations of motion estimation/compensation. Generally, when a video frame is scaled down, the motion activity for a macroblock will be changed. Two possible approaches can be used to derive the new motion vectors as well as the prediction errors.

A straightforward approach is to perform motion re-estimation after scaling operation. Note that the motion estimation is required to be executed in the spatial domain, which necessitate decoding of the MPEG video. In addition, both decoding and motion estimation have a high computational complexity. Hence, this approach is not suitable for applications which demand fast processing.

Another efficient approach to obtain the motion vectors and prediction errors for the scaled video is to derive them from the original motion vectors and prediction errors. Hence, the unnecessary decompression and motion re-estimation procedures can be avoided resulting in

fast execution. In this chapter, we propose a new approach to implement encoding of the scaled video directly in the MPEG motion-compensated DCT domain by using a combined FC-DCT and modified DCT domain motion compensation technique. A key step in the encoder is the macroblock based motion estimation/compensation for the scaled frames.

We note that a new macroblock in the scaled frame corresponds to four macroblocks in the original frame. It is therefore likely that several macroblocks in a large homogeneous area of a frame move in the same direction with similar velocities, and hence the motion information between adjacent macroblocks is highly correlated (Fig. 5.1). In this case, we can interpolate the motion vector for a macroblock in the scaled frame from the original motion vectors associated with four adjacent macroblocks in the original frame. However, it is possible that in certain cases, the four original motion vectors are not quite similar. For example, they may have different directions or large differences in velocities (Fig. 5.2). In this case, it is difficult to get a reasonable representation motion vector which reflects the motion activity of a macroblock in the scaled frame. In such a case, we propose to disregard the motion compensation process, and enforce the new macroblock to be intra-coded.

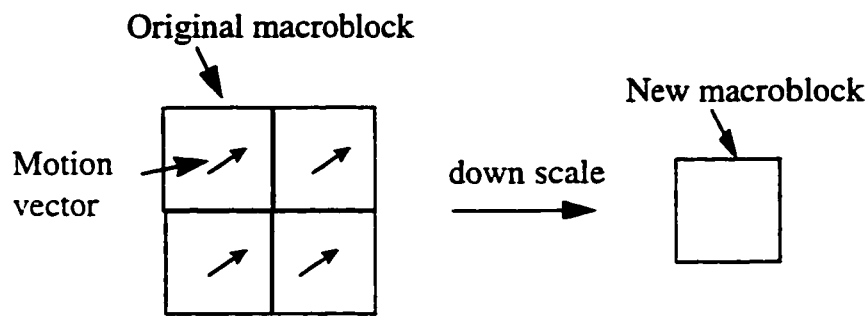


Fig. 5.1. Four original macroblocks with similar motion activity

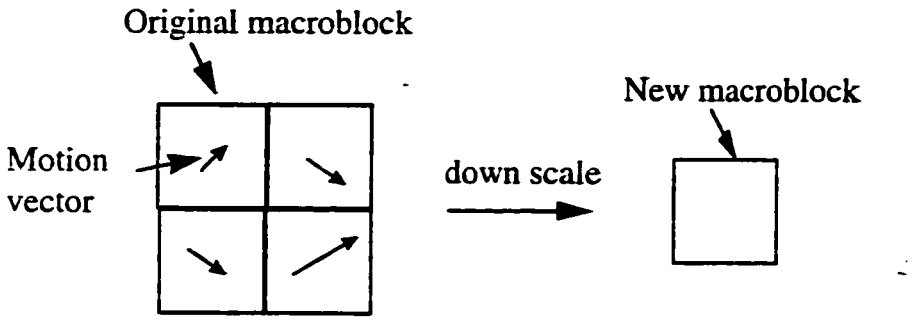


Fig. 5.2. Four original macroblocks with different motion activity

Hence, depending on the analysis of the motion activity, a new macroblock in the scaled frame can be re-encoded as either inter-coded or intra-coded. We call this encoder the macroblock type selective (MTS) DCT domain encoder, which performs all of the operations in the compressed domain. Most importantly, the proposed MTS-encoder is compatible with the standard MPEG encoder. Hence, it provides an elegant way to re-encode the scaled video while maintaining a superior video quality.

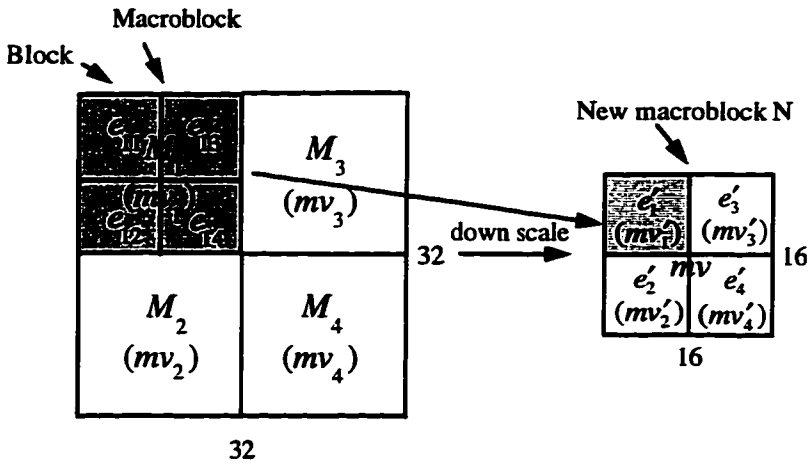
5.2 MTS-Encoder for Scaled MPEG Video

To start with, we need to partially decode the MPEG bit stream to get access to the block DCT coefficients and the motion vectors associated with the macroblocks.

For I-frames, motion estimation/compensation operations are not required. Therefore, the FC-DCT technique as discussed in section 4.2 for scaling an image in the DCT domain can be directly employed. The resulting scaled version of the block DCT coefficients will be stored for reconstruction of scaled I-frames as well as for further prediction for P- and B-frames.

For P- and B-frames, additional computations are required for motion compensation. We first present the approach for manipulation of P-frames, and will further extend it to B-frames.

To obtain a new macroblock N in the scaled frame, four adjacent macroblocks $M_1 \sim M_4$ in the original frames are considered. For the sake of simplicity, we use a gray level video sequence, where each macroblock consists of only four luminance blocks. As shown in Fig. 5.3(a), for the original macroblocks, each macroblock M_i $(i = 1,\cdots,4)$ is inter-coded, and thus having a motion vector $mv_i (i = 1,\cdots,4)$ and the corresponding four prediction error block data $e_{i1} \sim e_{i4}$ $(i = 1,\cdots,4)$. The DCT domain data for the prediction errors is DCT(e_{i1})~DCT(e_{i4}) $(i = 1,\cdots,4)$, which is called difference block DCT.



(a) Four adjacent original macroblocks (b) A new macroblock

Fig. 5.3. Down-scaling operation for P-frames

When the motion activity of adjacent original macroblocks are similar, the new scaled macroblock would still be inter-coded. As shown in Fig. 5.3(b), the upper-left prediction error block e'_1 of macroblock N in a scaled frame corresponds to $e_{11} \sim e_{14}$ of M_1 in the original

frame. Using a similar technique as for I-frames, the difference block DCT, i.e. DCT(e_i') ($i = 1,\cdots,4$) can be obtained by scaling $DCT(e_{i1}) \sim DCT(e_{i4})$ ($i = 1,\cdots,4$) in the DCT domain.

From Fig. 5.3, it can be seen that the original motion vectors $mv_1 \sim mv_4$ are scaled down to $mv_1' \sim mv_4'$. Because of the linearity of scaling operation, $mv_1' \sim mv_4'$ can be calculated using equation (5.1).

$$mv_i' = \frac{mv_i}{2} \qquad i = 1,\cdots,4 \tag{5.1}$$

We note $mv_1' \sim mv_4'$ represent motion of the component blocks in a macroblock. Therefore, the prediction error blocks $e_1' \sim e_4'$ correspond to the block based motion vectors. In order to re-encode the new macroblock in accordance with the MPEG standard, a single motion vector for a macroblock is required, while the prediction errors should be based on this new motion vector.

Here, we derive the new motion vector mv for a macroblock N by interpolating the scaled motion vectors $mv_1' \sim mv_4'$ using equation (5.2). Generally, the four motion vectors $mv_1 \sim mv_4$ are not identical, hence mv is different from $mv_1' \sim mv_4'$, which implies that $e_1' \sim e_4'$ are no longer accurate representations of the prediction errors between the current block and the motion compensated block based on mv. In order to apply mv to the new macroblock while maintaining a good reconstruction of the predictive frame, certain modifications on the prediction errors for each block are essential.

$$mv = \frac{mv_1' + mv_2' + mv_3' + mv_4'}{4} \tag{5.2}$$

We define the new prediction errors for the four blocks as $e_1 \sim e_4$, which correspond to the new motion vector mv. We now show how to obtain $DCT(e_1) \sim DCT(e_4)$.

Let the upper-left reconstructed block in macroblock N be represented by r_1. Using the block based motion vector mv_1', r_1 can be reconstructed by adding the prediction error block e_1' to the motion compensated block from the reference frame (equation (5.3)), where the reference frame can be either an I-frame or a P-frame in the scaled version.

$$r_1 = mc' + e_1' \tag{5.3}$$

where mc' is the motion compensated block using block based motion vector mv_1'.

Using DCT on each side of equation (5.3), the DCT coefficients of r_1 can be calculated as follows,

$$DCT(r_1) = DCT(mc') + DCT(e_1') \tag{5.4}$$

In equation (5.4), $DCT(e_1')$ is already available (by scaling $DCT(e_{11}) \sim DCT(e_{14})$), however, when the motion compensated block is not aligned with the structure of blocks, $DCT(mc')$ is not immediately available. We follow Chang and Messerschmitt's method [18] to reconstruct the DCT coefficients of a motion compensated block with an arbitrary position from the DCT coefficients of the reference frame.

Generally, in the scaled version of the reference frame, the motion compensated block overlaps with four blocks, where the DCT coefficients of the four blocks are already available. We note each motion vector has two components, namely horizontal (mx) and vertical (my), which provide an offset from the coordinate position in the current frame to the coordinate in the reference frame. Fig 5.4 shows a case when both mx and my are positive, i.e. the motion compensated block in the reference frame is in the lower-right position with respect to the current block.

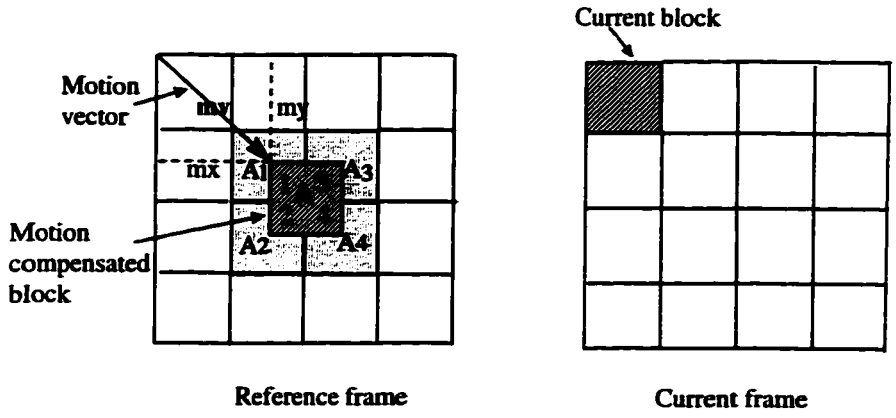


Fig. 5.4. A motion compensated block overlaps with four blocks ($mx>0$, $my>0$)

It can therefore be seen from Fig. 5.4, that a motion compensated block A consists of four parts, which are from neighboring blocks $A_1 \sim A_4$, respectively. Therefore, block A can be obtained by linear combination of blocks $A_1 \sim A_4$ using equation (5.5), where the translations of blocks $A_1 \sim A_4$ depend on the respective motion vector (mx, my).

$$A = \sum_{i=1}^{4} H_i A_i W_i \tag{5.5}$$

where $H_1 = H_3 = \begin{bmatrix} 0 & I_{8-(my)_8} \\ 0 & 0 \end{bmatrix}$ $\qquad H_2 = H_4 = \begin{bmatrix} 0 & 0 \\ I_{(my)_8} & 0 \end{bmatrix}$

$W_1 = W_2 = \begin{bmatrix} 0 & 0 \\ I_{8-(mx)_8} & 0 \end{bmatrix}$ $\qquad W_3 = W_4 = \begin{bmatrix} 0 & I_{(mx)_8} \\ 0 & 0 \end{bmatrix}$

I_n is an identity matrix with size nxn.

mx, my are the horizontal and vertical components of a motion vector.

$(mx)_8, (my)_8$ denote absolute values for mx and my modulo 8, respectively.

In equation (5.5), multiplying matrix A_i with a pre-matrix H_i and a post-matrix W_i provides manipulations of the rows and columns of matrix A_i.

We illustrate the calculation of A from one of its contributing blocks A_1 in Fig. 5.5.

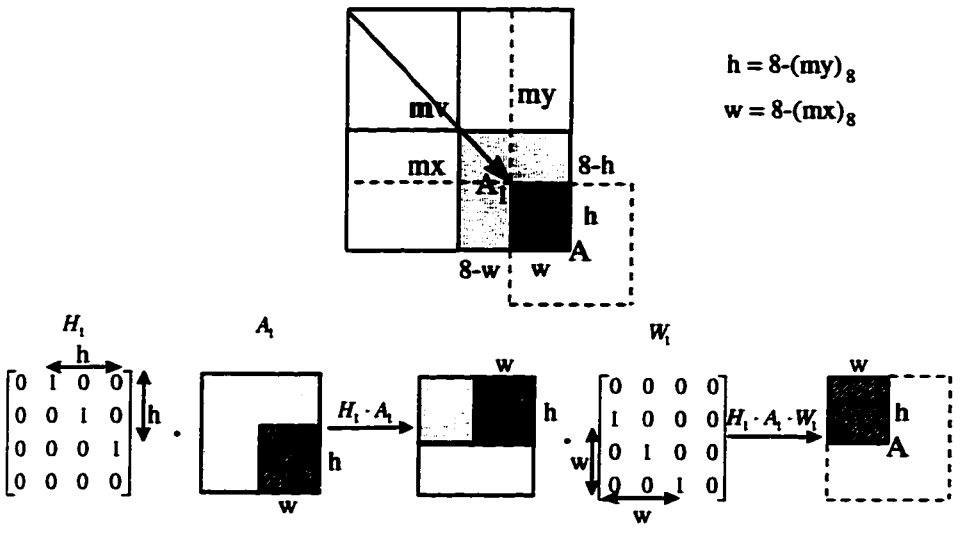


Fig. 5.5. The calculation of a motion compensated block

Based on equation (5.5), the DCT coefficients of the motion compensated block A can be derived as follows,

$$DCT(A) = \sum_{i=1}^{4} DCT(H_i)DCT(A_i)DCT(W_i) \tag{5.6}$$

In equation (5.6), when either mx or my is an integral multiple of the block width as shown in Fig. 5.6, only two blocks in the reference frame are involved in the calculation of the motion compensated block DCT coefficients. Furthermore, if the motion compensated block is aligned exactly with the structure of blocks (Fig. 5.7), its DCT coefficients can be obtained directly. For the other cases as shown in Fig. 5.8-5.10, when ($mx>0$, $my<0$), ($mx<0$, $my>0$), and ($mx<0$, $my<0$), similar formulas can be derived for calculating the motion compensated block DCT coefficients.

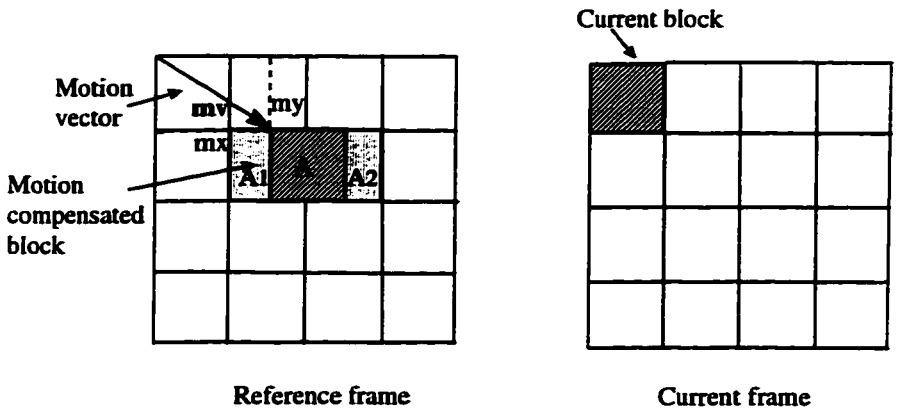


Fig. 5.6. A motion compensated block overlaps with two blocks

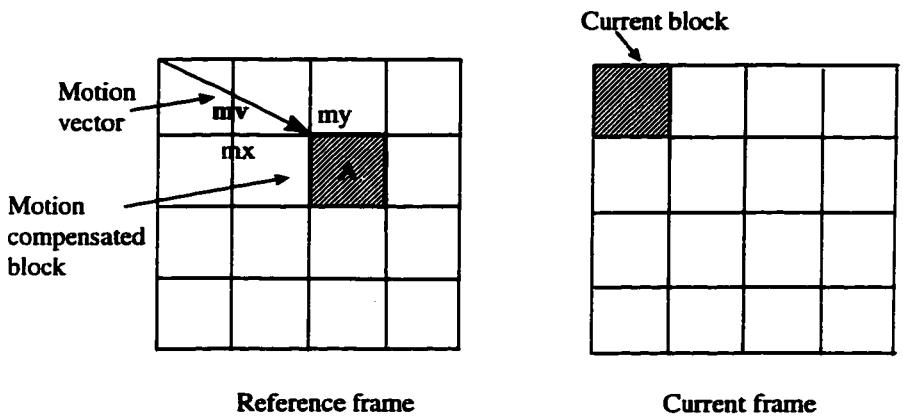


Fig. 5.7. A motion compensated block is aligned with the structure of blocks

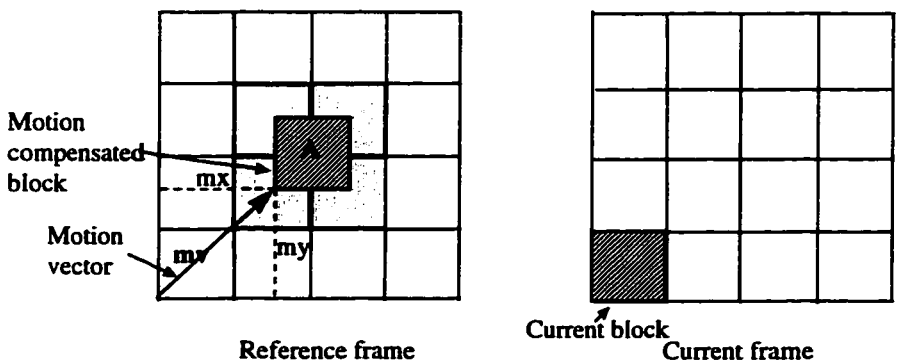


Fig. 5.8. A motion compensated block overlaps with four blocks ($mx>0$, $my<0$)

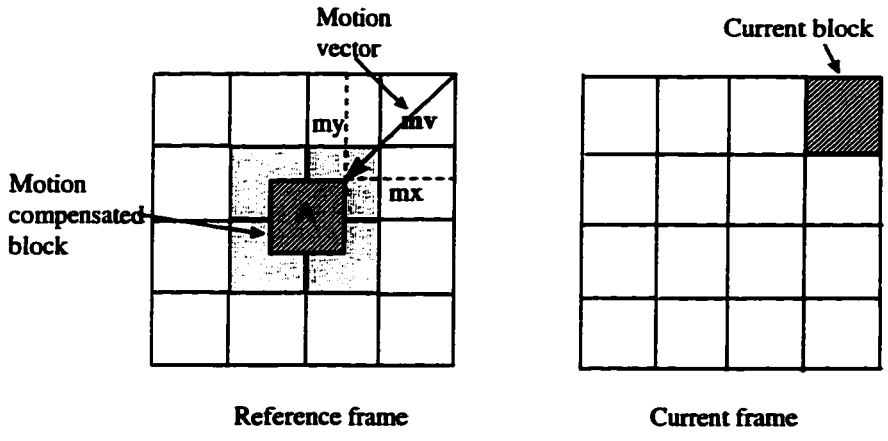


Fig. 5.9. A motion compensated block overlaps with four blocks ($mx<0$, $my>0$)

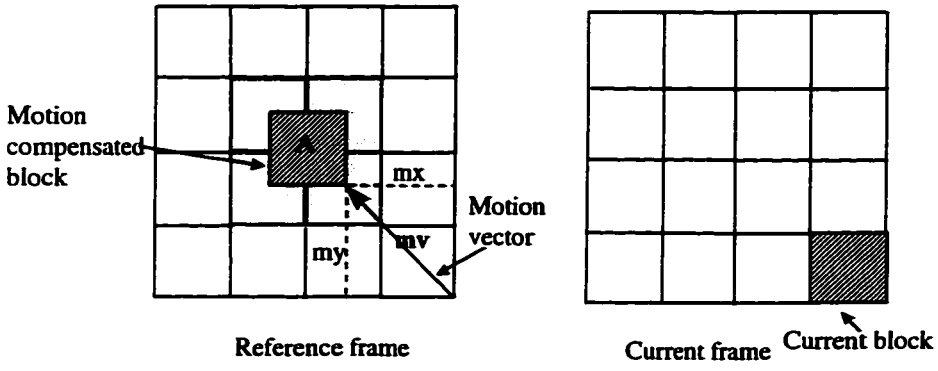


Fig. 5.10. A motion compensated block overlaps with four blocks ($mx<0$, $my<0$)

In other words, the motion compensated block DCT coefficients can be obtained from the existing block DCT coefficients. This technique can therefore be used to calculate the item $DCT(mc')$ in equation (5.4), where $DCT(mc')$ is based on block motion vector mv_1'. In summary, the reconstructed block DCT coefficient DCT(r_1) can be obtained directly in the DCT domain.

Alternatively, DCT(r_1) can also be reconstructed using equation (5.7), when using macroblock based motion vector mv and the respective prediction error e_1. By comparing

equations (5.4) and (5.7), it is clear that the macroblock motion based $DCT(e_1)$ can be derived using equation (5.8). It can be observed that the modifications on $DCT(e_1)$ is only related to the difference of the two motion-compensated blocks based on motion vectors mv_1' and mv, respectively.

$$DCT(r_1) = DCT(mc) + DCT(e_1) \tag{5.7}$$

$$DCT(e_1) = DCT(e_1') + \left(DCT(mc') - DCT(mc)\right) \tag{5.8}$$

where $DCT(mc)$ is based on macroblock motion vector mv.

$DCT(mc') - DCT(mc)$ is the modifications on $DCT(e_1')$.

For the other three blocks (r_1, r_2, r_3) in macroblock N, the same method can be used to obtain the difference block DCT. After these calculations, both the motion vectors for macroblocks and block DCT of prediction errors in the P-frames are available for entropy coding. Note that we need to store the reconstructed block DCT coefficients $DCT(r_i)$ $(i = 1, \cdots, 4)$, since P-frames are used as reference for successive P- and B-frames.

We note that the above discussion for deriving the new motion vectors from existing motion vectors is based on the assumption that the motion activity in the four adjacent macroblocks are similar. However, when the motion vectors are different, further processing is required. Irrespective of the difference in the values of the motion vectors, the modified prediction errors can still be obtained using equation (5.8). However, when the prediction errors are too large, the motion compensation is not effective for compression. The best solution is to enforce an intra-coded macroblock instead of inter-coded. We note that in the MPEG standard encoder, selection of inter- or intra-coded macroblock is dependent on the

effectiveness of motion estimation. We will discuss the decision making strategy (based on motion vectors) in section 5.3.

To obtain the motion vectors and the difference block DCT coefficients for the scaled B-frames, similar techniques as for P-frames can be used. Since both forward and backward predictions are required for B-frames, there are two motion vectors namely forward (mv_f) and backward (mv_b) for each macroblock. The block based motion vectors mv'_{fi} and mv'_{bi} ($i = 1,\cdots,4$) can be calculated using equation (5.9.a) and (5.9.b), respectively. Meanwhile, the macroblock based motion vectors mv_f and mv_b can be derived by interpolating the scaled motion vectors as shown in equation (5.10.a) and (5.10.b), respectively.

$$mv'_{fi} = \frac{mv_{fi}}{2} \qquad i = 1,\cdots,4 \tag{5.9.a}$$

$$mv'_{bi} = \frac{mv_{bi}}{2} \qquad i = 1,\cdots,4 \tag{5.9.b}$$

$$mv_f = \frac{mv'_{f1} + mv'_{f2} + mv'_{f3} + mv'_{f4}}{4} \tag{5.10.a}$$

$$mv_b = \frac{mv'_{b1} + mv'_{b2} + mv'_{b3} + mv'_{b4}}{4} \tag{5.10.b}$$

As shown in equation (5.4), the reconstruction of the block DCT ($DCT(r_1)$) in a scaled macroblock for B-frames involves the addition of the difference block DCT ($DCT(e'_1)$) with the motion compensated block DCT ($DCT(mc')$). However, some modifications are required for B-frames, in that the motion compensated block mc' is formed by averaging the forward and backward motion compensated blocks from the previous and future reference frames, respectively. Hence $DCT(mc')$ can be calculated using equation (5.11).

$$DCT(mc') = \frac{DCT(mc'_f) + DCT(mc'_b)}{2} \tag{5.11}$$

Similarly, $DCT(r_1)$ can be reconstructed based on macroblock motion vectors using equation (5.7), where $DCT(mc)$ is obtained as follows,

$$DCT(mc) = \frac{DCT(mc_f) + DCT(mc_b)}{2} \tag{5.12}$$

From equations (5.4), (5.7), (5.11) and (5.12), the modified difference block DCT based on macroblock motion vector can be expressed as shown in equation (5.13).

$$DCT(e_1) = DCT(e'_1) + \left(\frac{DCT(mc'_f) + DCT(mc'_b)}{2} - \frac{DCT(mc_f) + DCT(mc_b)}{2} \right) \tag{5.13}$$

where $\frac{DCT(mc'_f) + DCT(mc'_b)}{2} - \frac{DCT(mc_f) + DCT(mc_b)}{2}$ is the modification on $DCT(e'_1)$.

It can be seen that for a scaled B-frame, both the forward and backward motion vectors as well as difference block DCT can be derived in the motion-compensated DCT domain for entropy coding. Furthermore, we can also enforce an intra-coded macroblock whenever motion compensation is not effective.

In the above discussions, we have presented an approach for re-encoding scaled I-, P- and B-frames in the motion-compensated DCT domain. From equations (5.7) and (5.8), it can be observed that the derivation of the difference block DCT coefficients ($DCT(e_i)$) requires the reconstruction of the scaled block DCT coefficients ($DCT(r_i)$). We note that the reconstruction of $DCT(r_i)$ also provides an approach to scale down a frame in the motion-compensated DCT domain (before decoding). This can be used in decoding applications, where the procedures for obtaining new motion vectors and difference block DCT coefficients are not required. To demonstrate the validation of the scaling operations as well as the re-encoding

approach, simulations are performed using three test (MPEG) video sequences: "Garden", "Tennis" and "Football", which represent low motion, medium motion and high motion, respectively. We have shown the reconstructed scaled "Garden", "Tennis" and "Football" sequences in Fig. 5.11.1-5.11.3, respectively. The DCT domain scaling technique is equivalent to the spatial domain technique. From the computational complexity point of view, it is especially useful for re-encoding the scaled video sequences.

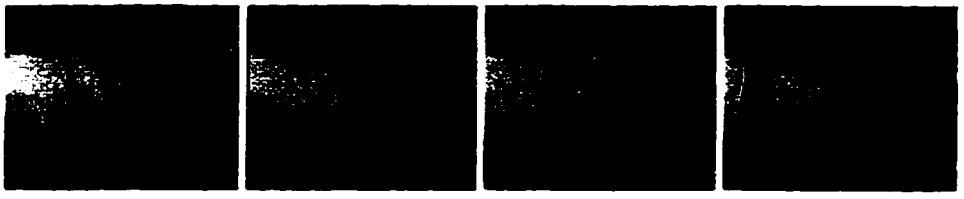


Fig. 5.11.1. "Garden" sequence scaled by DCT domain technique

Fig 5.11.2. "Tennis" sequence scaled by DCT domain technique

Fig. 5.11.3. "Football" sequence scaled by DCT domain technique

5.3 Macroblock Type Selection Strategy in MTS-Encoder

5.3.1 Motion Vectors for Macroblock Type Selection

The decision for macroblock type (inter/intra) selection is an important step in the MTS-encoder. Based on the effectiveness of motion compensation for the scaled macroblocks, intra-coded macroblocks may be introduced for P- and B-frames, which is proposed in the MPEG standard.

Recall from section 2.6, that the inter/intra coding decision for P- and B-frames in the MPEG algorithm is based on the variances of the luminance component of a macroblock. The decision for a macroblock in P-frames is as follows:

$$\begin{cases} \text{if } V_d \le 64 \text{ or } V_d < V_c & \text{macroblock is inter}-\text{coded} \\ \text{otherwise} & \text{macroblock is intra}-\text{coded} \end{cases} \tag{5.14}$$

where V_d is the variance of the difference macroblock.

V_c is the variance of the current macroblock.

We note that when we calculate V_d, the average value is assumed to be zero. Therefore, small values of V_d always indicate the effectiveness of prediction.

Recall from section 5.2 that, in the MTS-encoder, both the current macroblock and the difference macroblock data are available in the form of DCT coefficients. If the encoder selects macroblock type by using the variances, decoding of the DCT coefficients will be involved. An alternative approach is to take advantage of the existing motion vectors from the coded bit stream. By analyzing the underlying relationship between motion vectors and variances of the current and difference macroblocks, we propose to chose macroblock coding type by using the motion vectors information.

5.3.2 Experimental Results and Analysis

To calculate variances of macroblocks, we execute inverse DCT to recover the current macroblock and the difference macroblock data in the spatial domain.

First of all, we investigate the possibility of enforcing intra-coded macroblocks after the scaling operation. Let p_f denote the probability of enforcing an intra-coded macroblock, and $1-p_f$ denote the probability that the scaled macroblock is inter-coded, where p_f is calculated on a frame by frame basis. Fig 5.12.1-5.12.3 have shown the statistical probability p_f for 45 sampled P-frames from the scaled version of "Garden", "Tennis" and "Football" sequences, respectively.

It can be seen that for low motion video sequences, p_f is generally less than 0.1, while for medium and high motion video sequences, p_f is generally between 0.2 and 0.5. Therefore, for medium and high motion sequences, it is more likely that some of the macroblock coding types are modified after scaling operation.

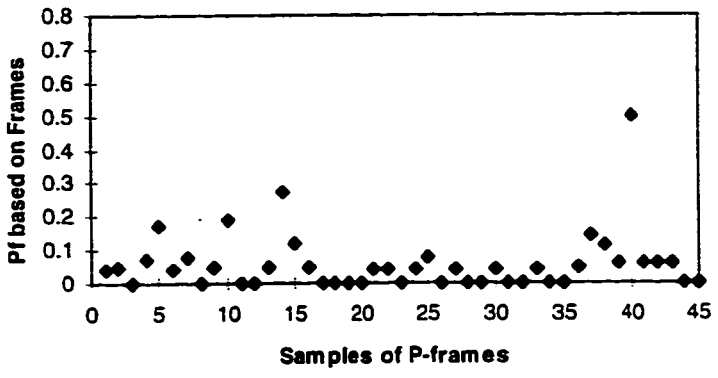


Fig. 5.12.1. Probability p_f for "Garden" sequence

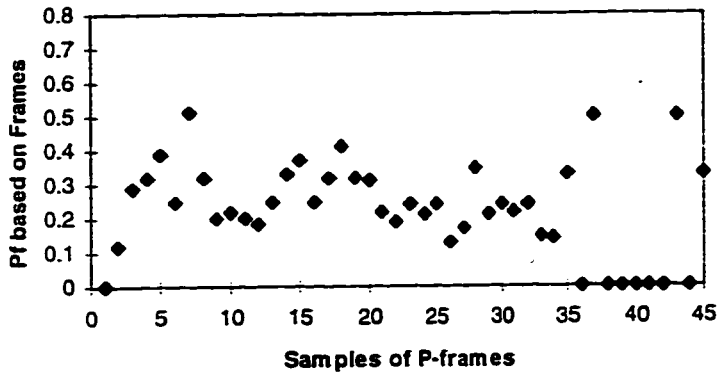


Fig. 5.12.2. Probability p_f for "Tennis" sequence

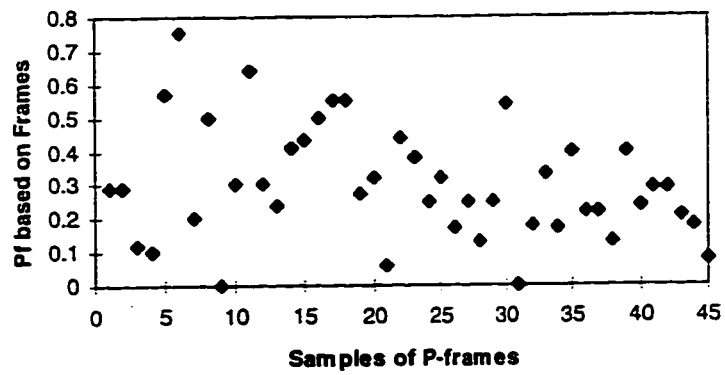


Fig. 5.12.3. Probability p_f for "Football" sequence

Next, we analyze the motion vector information for macroblock type selection. We recall from section 5.2 that a new motion vector mv is interpolated from the block based neighboring motion vectors $mv_1' \sim mv_4'$. Let the difference motion vector d_m denote the sum of distance between mv and $mv_1' \sim mv_4'$. For the sake of simplicity, we use absolute distance measure as shown in equation (5.15) to calculate d_m, since it gives a similar performance as that using Euclidean distance [40].

$$d_m = \sum_{i=1}^{4} \left(|mx - mx_i'| + |my - my_i'| \right) \tag{5.15}$$

where mx, my are the horizontal and vertical components of mv.

mx_i', my_i' are the horizontal and vertical components of mv_i'.

The range of d_m is dependent on the motion activity of the video sequence as well as the search range, which is determined by the encoder when encoding the original video sequence. The probability distribution $P(d_m)$ for each test sequence is graphed in Fig. 5.13.1-5.13.3, respectively.

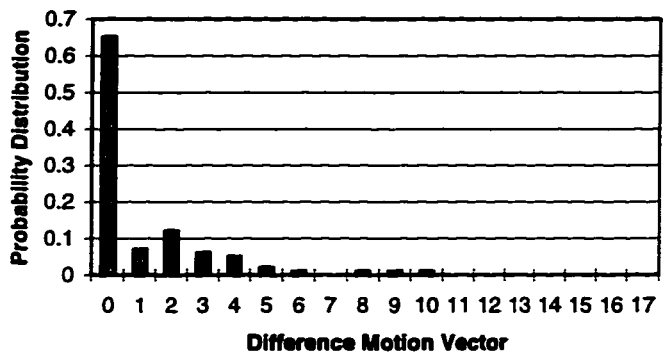


Fig.5.13.1. Probability distribution $P(d_m)$ for "Garden" sequence

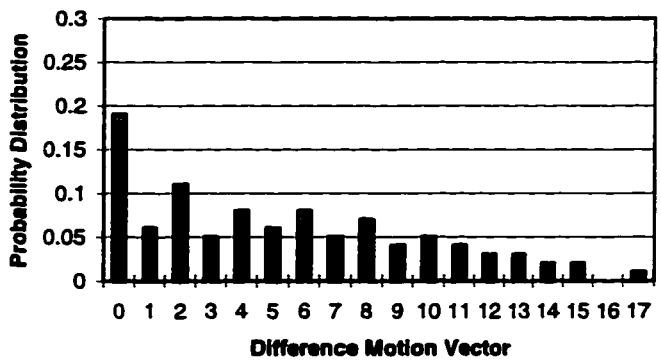


Fig.5.13.2. Probability distribution $P(d_m)$ for "Tennis" sequence

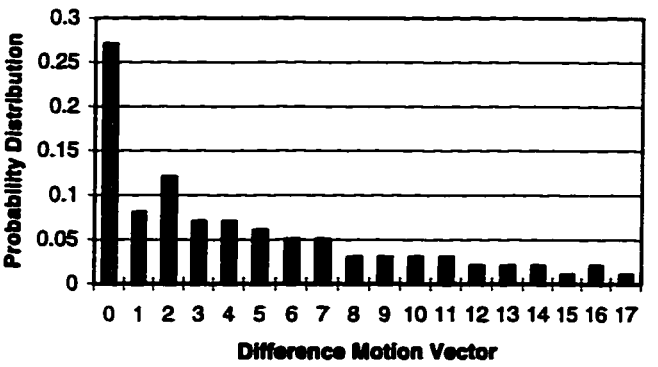


Fig.5.13.3. Probability distribution $P(d_m)$ for "Football" sequence

It can be seen that small values of d_m generally have high probability, especially for $d_m = 0$, which is the case when motion vectors $mv_1' - mv_4'$ are similar. Furthermore, medium and high motion video sequences are more likely to have large d_m values than low motion video sequences.

For the scaled sequences, a small value of d_m indicates an effective motion compensation, thus resulting in a relatively small value of V_d. On the contrary, when the magnitude of d_m becomes larger, motion compensation becomes less effective, hence it is very likely that the magnitude of V_d tends to be larger than V_c. In Fig. 5.14, we illustrate the relationship between inter/intra-coding selection and the difference motion vector for the three test sequences.

The X axis indicates the d_m value, while Y axis shows the probability p_m of enforcing a scaled macroblock to be intra-coded for that d_m value, which is calculated on a sequence basis. We note that in low motion sequences, few macroblocks have large d_m values, hence the corresponding probability p_m is not applicable. For example, the probability p_m for the "Garden" sequence is calculated for $d_m \leq 5$. It can be expected that, larger values of d_m usually

correspond to larger values of p_m. Hence, enforcing an intra-coded macroblock may be required for these cases.

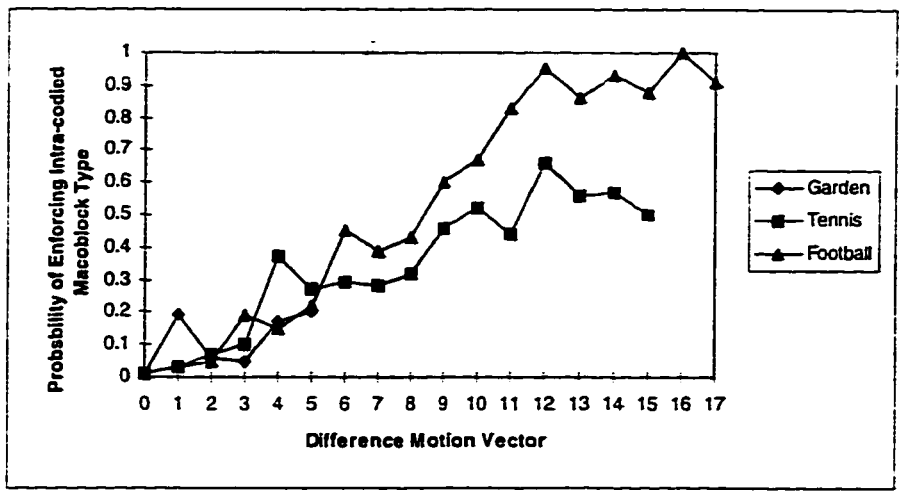


Fig. 5.14. Probability p_m vs. d_m for three test sequences

For low motion sequence ("Garden" in Fig. 5.14), the value of d_m is relatively small, in turn, the probability p_m for each d_m value is small (less than 0.2). Therefore, all scaled macroblocks can still be inter-coded. For medium and high motion sequences, we set global thresholds T_m on the value of d_m to determine the macroblock coding type. From the plots for "Tennis" and "Football" in Fig. 5.6, we observe that, with the increase of d_m, the probability p_m tends to increase as well, however, some degree of fluctuation exists.

In addition, it can be seen that, when d_m is large, the "Football" sequence has higher probability p_m compared to the "Tennis" sequence. Hence, in practice, different thresholds T_m may be chosen for video sequences with medium and high motion. Based on Fig. 5.6, if the intra-coding type is chosen when the probability $p_m > 0.5$, then thresholds $T_m = 12$ and $T_m = 9$

can be used for "Tennis" and "Football" sequences, respectively. We note when $p_m > 0.5$, it indicates that the scaled macroblock is more likely to be intra-coded due to the ineffectiveness of motion compensation.

In this approach, the global thresholds T_m is chosen based on statistical probability, hence, in some cases, incorrect decisions for macroblock coding type may be made. However, this will only introduce marginal reduction in the compression ratio without degrading the visual quality.

5.4 Summary

In this chapter, we have proposed an approach to scale and re-encode the MPEG video directly in the motion-compensated DCT domain. The proposed MTS-encoder removes the unnecessary decompression/re-compression and motion estimation procedures, and derives the motion information and DCT coefficients of the prediction errors for the scaled macroblocks from the original motion vectors and DCT coefficients, respectively.

A key step in the encoder is the macroblock based motion compensation for the scaled frames. By analyzing the motion activity, a scaled macroblock (corresponding to inter-coded macroblocks with original size) can be either inter-coded or intra-coded. The decision strategy is studied by using three test video sequences with different motions, i.e. low, medium and high. Simulation results have shown that, for low motion sequences, the macroblocks can be maintained to be inter-coded, while for medium and high motion sequences, different global thresholds T_m on the difference motion vector d_m can be used for choosing macroblock coding type.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

The recent advances in DSP, VLSI and high speed networking techniques have made video services increasingly popular. The large volumes of visual information for storage and transmission necessitate the use of compression. In advanced visual services, further manipulations of compressed visual information may be required. Spatial scalability has several applications including browsing visual databases, querying multimedia databases, and interactive multimedia communications. In addition, scaling multiple video sources and compositing them into a single representation is especially important for a video conferencing system.

In this thesis, we have investigated techniques to manipulate the visual data in the compressed domain, with particular focus on spatial scalability and scaling operations. First, we propose two novel techniques, namely format compatible (FC) DCT and format modified (FM) DCT to implement image/video spatial scalability directly in the DCT compressed

domain. These techniques remove the unnecessary decompression and re-compression procedures compared to the traditional spatial domain approaches, and therefore have a great potential in reducing computational complexity and storage requirements.

In the FC-DCT technique, the majority of the computations is to perform down-sampling and up-sampling operations on the image DCT coefficients, which essentially involve only pre-matrix and post-matrix multiplications. Based on the FC-DCT technique, we have proposed a DCT domain JPEG hierarchical mode encoder, which is functionally equivalent to the JPEG spatial domain technique. In the FM-DCT technique, the computational complexity is greatly reduced by using certain approximations. We note that depending on the image/video content, only marginal quality degradation (almost unnoticeable subjectively) may be introduced.

Simulation results confirm that the proposed DCT domain techniques can achieve a comparable performance at a much lower computational complexity compared to the spatial domain techniques. In addition, the FM-DCT has a significantly lower computational complexity at a comparable image quality with the FC-DCT technique. We note that the choice of the technique is based on the requirements of the specific application. For example, the FC-DCT technique can be used to manipulate the standard bit streams, while the FM-DCT technique can be employed for fast processing of visual data.

We note that the FC-DCT technique is used for manipulating and maitaining the visual data in the standard formats. However, it is not directly applicable to the MPEG video due to the motion estimation/compensation in the temporal dimension. Therefore, we propose a new approach to implement encoding of the scaled video directly in the MPEG motion-compensated DCT domain by using a combined FC-DCT and modified DCT domain motion compensation technique.

The proposed macroblock type selective (MTS) DCT domain encoder removes the unnecessary decompression/re-compression and motion re-estimation procedures, which are highly computational-intensive. The new motion information for the scaled macroblocks can be obtained by processing the original motion vectors. Depending on the difference motion activities for the original neighboring macroblocks, a new scaled macroblock can be either still inter-coded or enforced to be intra-coded.

In the MPEG algorithm. The inter/intra coding decision for a macroblock is based on the variances of the current macroblock and of the difference macroblock. In order to avoid the inverse DCT operation, the underlying relationship between motion vectors and variances of macroblocks has been studied. Simulation results have shown that for low motion sequences, all the macroblocks can be maintained to be inter-coded, while for medium and high motion sequences, different global thresholds T_m on the difference motion vector d_m can be used for choosing macroblock coding type. The proposed MTS-encoder is also compatible with the standard MPEG encoder.

6.2 Future Work

In this thesis, we have provided the DCT domain scaling operation to scale an image/video frame by half. The extension of it is fractional size scaling operation. For a video conferencing system, it is very desirable to scale multiple video sources to arbitrary sizes. We note that DCT is a unitary orthonormal transform, and it is distributive to matrix multiplication. Therefore. if in the spatial domain, the manipulation can be implemented by linear combination of matrix multiplication, it is always possible to derive the equivalent DCT domain operation.

We note that the proposed MPEG compressed domain scaling algorithm is implemented on the luminance component of a frame. It can be easily extended for the two chrominance components as well. In the MPEG standard, a macroblock has four luminance blocks (Y) and two chrominance blocks (Cr and Cb). Since motion estimation is based on the luminance information only, all blocks share the same motion vector. To reconstruct a motion compensated block for luminance in the DCT domain, two approaches can be used. In the first approach, a reference macroblock is first scaled down to a block, followed by a DCT domain motion compensation operation on this block. In the second approach, the order of scaling and motion compensation operations are swapped. From an implementation point of view, these two approaches are equivalent. However, the latter approach requires motion compensation for four blocks (in a macroblock), thus increases the computational load.

In the current implementation, we use the former approach. However, if we extend the encoder to process chrominance components, the latter approach must be chosen. This is because the four blocks which form one block in a scaled macroblock are in different macroblocks (with different motion vectors), therefore, motion compensation must be executed individually for each block.

In the proposed MTS-encoder, different global thresholds on the difference motion vector for video sequences with different motion classifications are used for macroblock type selection. This relies on users to identify the general motion information in a video sequence, and guide the encoder to use an appropriate threshold. The performance and operability can be improved if the encoder can analyze the local motion activity and further determine an appropriate local threshold.

The proposed algorithm for encoding the scaled MPEG video in the compressed domain can be extended to multiple layers for implementation of spatial scalability, which can be used to update an MPEG-1 video to an MPEG-2 spatial scalability video. Unlike the JPEG spatial scalability, where only spatial prediction is used for encoding of the enhancement layer, the MPEG-2 spatial scalability algorithm employs two kinds of predictions. First, a temporal prediction is made from the previous frames in the enhancement layer. Second, a spatial prediction is formed from the up-sampled version of the lower layer frame.

Finally, integration of scaling and scalable coding of image/video in the compressed domain into multimedia database for fast processing applications opens up many new areas of future research work. Examples include compressed domain visual data editing/publishing, visual data browsing, retrieval and indexing.

Bibliography

[1] ISO/IEC/JTC1/SC2/WG8 10918-1 (JPEG), "Information Technology- Digital Compression and Coding of Continuous-tone Still Images", Committee Draft, 1991.

[2] G. K. Wallace, "The JPEG Still Picture Compression Standard", *Communications of the ACM*, vol. 34, no. 4, pp. 30-44, Apr. 1991

[3] D. Le Gall, "MPEG: A video compression standard for multimedia applications", *Communications of the ACM*, vol. 34, no.4, pp. 46-58, April 1991.

[4] ISO/IEC JTC1/SC29/WG11 13818-2 (MPEG-2), "Information technology - generic coding of moving pictures and associated audio", Committee Draft, November 1993 .

[5] M. Liou, "Overview of the px64 kbits/s video coding standard", *Communications of the ACM*, vol. 34, no. 4, pp. 59-63, April 1991.

[6] C. Gonzales and E. Viscito, "Flexibly Scalable Digital Video Coding", *Signal Processing: Image Communications*, vol. 5, pp. 5-20, 1993.

[7] B. Girod, "Scalable Video for Multimedia Workstations", *Computer & Graphics*, Pergamon Press Ltd., vol. 17, no. 3, pp. 269-276, 1993.

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

[9] T. Hanamura, W. Kameyama and H. Tominaga, "Hierarchical Coding Scheme of Video Signal with Scalability and Compatibility", *Signal Processing: Image Communications*, vol. 5, pp. 159-184, 1993.

[10] A. Puri and A. Wong, "Spatial Domain Resolution Scalable Video Coding", *Visual Communications and Image Processing '93, SPIE*, vol. 2094, part 1, pp. 718-729, Cambridge, Massachussetts, Nov. 1993.

[11] P. Boucher and M. Goldberg, "Transform Image Coding by Vector Quantization", *Ninth Symp. On Signal Processing and Applications*, pp. 629-633, Nice France, May 1983.

[12] A. Tran, K. M. Liu, K. H. Tzou and E. B. Vogel, " An Efficient Pyramid Image Coding System", *IEEE Int. Conf. On ASSP, Dallas, Texas*; pp. 744-747, Apr. 1987.

[13] P. J. Burt and E. H. Adelson, "The Laplacian Pyramid as a Compact Image Code", *IEEE Transaction on Communications*, vol. COM-31, pp. 532-540, Apr. 1983.

[14] T. H. Wendler and D. M. Ebrecht, "Proposed Standard for Variable Format Picture Processing and a Codec Approach to Math Diverse Imaging Devices", *Picture Archiving and Communications Systems (PACS) for Medical Applications, SPIE*, vol. 318, part 1, pp. 298-305, 1982.

[15] B. C. Smith and L. Rowe, "Algorithms for Manipulating Compressed Images," *IEEE Computer Graphics and Applications*, pp. 34-42, Sept. 1993.

[16] B. C. Smith, "Fast Software Processing of Motion JPEG Video", *Multimedia '94 Conference*, San Francisco, CA, pp. 77-88, Oct. 1994.

[17] S. Chang, W. Chen and D. G. Messerschmitt, "Video Compositing in the DCT Domain," *IEEE Workshop on Visual Signal Processing Communications*, Raleigh, NC, pp. 138-143, Sept. 1992.

[18] S. Chang and D. G. Messerschmitt, "Manipulation and Compositing of MC-DCT Compressed Video," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 1, pp. 1-11, Jan. 1995.

[19] S. Chang and D. G. Messerschmitt, "Compositing Motion-Compensated Video within the Network", *IEEE 4th Workshop Multimedia Communications*, Monterey, CA, pp. 40-56, Apr. 1992.

[20] S. Chang and D. G. Messerschmitt, "A New Approach to Decoding and Compositing Motion Compensated DCT-based Images", *IEEE ICASSP*, Minneapolis, MN, vol. 5, pp. v421-424, Apr. 1993.

[21] B. K. Natarajan and B. Vasudev, "A Fast Approximate Algorithm for Scaling Down Digital Images in the DCT Domain", *IEEE Int. Conf. on Image Processing (ICIP)*, vol. 2, pp. 241-243, Oct. 1995.

[22] W. H. Chen, C. H. Smith, and S. C. Fralick, "A fast computational algorithm for the discrete cosine transform", *IEEE Transactions on Communications*, vol. COM-25, no. 9, pp. 1004-1009, Sept. 1977.

[23] J. R. Jain and A. K. Jain, "Displacement Measurement and Its Application in Interframe Image Coding", *IEEE Transactions On Communications*, vol. COM-29, no. 12, pp. 1799-1808, Dec. 1981.

[24] A. N. Netravali and J. D. Robbins, "Motion Compensated Television Coding: Part 1", *Bell Syst. Tech. J*, vol. 58, pp. 631-670, Mar. 1979.

[25] D. Chen and A. C. Bovik, "Visual Pattern Image Coding". *IEEE Transaction on Communications*, vol. 38, no. 12, pp. 1662-1671, Dec. 1990.

[26] R. Steinmetz, " Data Compression in Multimedia Computing - Principles and Techniques", *Multimedia System'94*, vol. 1, pp. 166-172, Feb. 1994.

[27] G. G. Langdon, "An Introduction to Arithmetic Coding", *IBM J. Res. Develop*, vol. 28, no. 2, pp. 135-149, Mar. 1994.

[28] J. Rissanen and G. G. Langdon, "Arithmetic Coding", *IBM J. Res. Develop.*, vol. 3, no. 2, Mar. 1979.

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

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

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

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

[33] K. R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications*, Academic Press, London, 1990.

[34] C. T. Chen, "Video Compression: Standards and Applications", *Journal of Visual Communications and Image Representation*, vol. 4, no. 2, pp. 103-111, Jun. 1993.

[35] J. W. Woods and S. D. Oneil, "Subband Coding of Images", *IEEE Transaction, Acous. Speech Signal processing*, ASSP-34(5), pp1278-1288, 1986.

[36] S. G. Mallat, "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation", *IEEE Transaction on Pattern Anal. And Mach. Intel.*, vol. 11, no. 7, Jul. 1989.

[37] M. Antonini, M. Barlaud, P. Mathieu and I. Daubechies, "Image Coding using Wavelet Transform", *IEEE Transaction on Image Processing*, vol. 1, no. 2, Apr. 1992.

[38] N. M. naerabadi and R. A. King, "Image Coding using Vector Quantization: A Review", *IEEE Transaction on Communications*, vol. COM-36, no. 8, pp. 957-971, Aug. 1988.

[39] Y. Linde, A. Buzo and R. Gray, " An Algorithm for Vector Quantizer Design", *IEEE Transaction on Communications*, vol. COM-28, pp. 84-95, Jan. 1980.

[40] A. N. Netravali and J. O. Limb, "Transform Picture Coding", *Proceedings of the IEEE*, vol. 68, no. 3, pp. 366-407, Mar. 1980.

[41] J. C. Candy, M. A. Franke, M. A. Haskel and F. W. Mounts, "Transmitting Television as Clusters of Frame to Frame Differences", *Bell Syst. Tech. J.*, vol. 50, no. 6, pp. 1889-1917, 1971.

[42] W. K. Pratt, *Digital Image Processing*, Academic Press, New York, 1985.

[43] H. G. Musmann, O. Pirsch and H. J. Grallert, "Advances in Picture Coding", *Proc. of the IEEE*, vol. 73, no. 4, pp. 523-548, Apr. 1985.

[44] A. N. Netravali and B. G. Haskell, *Digital Pictures*, Plenum, New York, 1989.

[45] J. Biemond, L. Looijenga, D. E. Boekee and R. H. J. M. Plompen, "A Pel-recursive Wiener-based Displacement Estimation Algorithm", *Signal Processing*, vol. 13, no. 4, pp. 399-412, 1987.

[46] R. J. Moorhead II, S. A. Rajala and L. W. Cook. "Image Sequence Compression using A Pel-recursive Motion-compensated Technique", *IEEE Journal on Selected Areas in Communications*, vol. 5, no. 7, pp. 1100-1114, Aug. 1987.

[47] T. koga, K. Linuma, A. Hirano, Y. lijima and T. Ishiguro, "Motion-Compensated Interframe Coding for Video Conferencing", *Proceedings of the National Telecommunications Conference'81*, pp. G5.3.1-G5.3.5, 1981.

[48] M. Ghanbari, "The Cross-search Algorithm for Motion Estimaiton", *IEEE Transactions on Communicaitons*, vol. COM-38, no. 7, pp. 950-953, Jul. 1990.

[49] W. B. Pennebaker and J. L. Mitchell, JPEG Still Image Data Compression Standard, Yan Nostrand Reinhold, New York, 1993.

[50] R. A. Devore, B. Jawerth and B. J. Lucier, "Image Compression through Wavelet Coding," *IEEE Trans. on Information Theory*, March 1992.

[51] M. Ohta, M. Yana and T. Nishitani, "Entropy Coding for Wavelet Transform of Image and its Applications for Motion Picture Coding", *SPIE Visual Communications and Image Processing'91*, pp. 456-466, 1991.

[52] W. D. Ray and R. M. Driver, "Further Decomposition of the Karhunen Loeve Series Representation of a Stationary Random Process", IEEE Transactions on Information Theory, vol. IT-16, no. 6, pp. 663-668, Nov. 1970.

[53] R. Clarke, *Transform Coding of Images*, Academic Press, New York, 1985.

[54] A. S. Lewis and G. Knowles, "Image Compression using the 2-D Wavelet Transform", *IEEE Transactions on Image Processing*, vol. 1, no. 2, pp. 244-250, Apr. 1992.

[55] A. S. Lewis and G. Knowles, "Video Compression using 3D Wavelet Transform", *Electronic Letter*, vol. 26, no. 6, pp. 396-397, 1990.

[56] ISO/IEC JTC1 11172-2 (MPEG-1), "Information Technology - Coding of Moving Pictures and Associated Audio for Digital Storage Media up to about 1.5Mbits/s", Committee Draft, 1993 .

[57] ITU-T Recommendation H.261, "Video Codec for Audiovisual Services at px64 kbits/s", Geneva, 1990.

[58] ITU-T Recommendation H.263, "Video Coding for Narrow Telecommunication Channels at <64 kbits/s", Apr. 1995.

[59] ITU-T Recommendation H.263, "Video Coding for Low Bitrate Communication (TMN5)", Jul. 1995.

[60] N. Farber, E. Steinbach and B. Girod, "Robust H.263 Compatible Video Transmission over Wireless Channels", *Proc. International Picture Coding Symposium (PCS)*, vol. 2, pp. 575-578, Melbourne, Australia, Mar. 1996.

[61] R. Steinmetz, "Data Compression in Multimedia Computing - Standards and Systems", *Multimedia Systems'94*, vol. 1, pp. 187-204, 1994.

[62] Y. Altunbasak and A. M. Tekalp, "An H.263-compatible System for Video Coding based on a Triangular-mesh", *Proc. International Picture Coding Symposium (PCS)*, vol. 1, pp. 129-133, Melbourne, Australia, Mar. 1996.

[63] ISO/IEC JTC1/SC29/WG11, "First Draft of MPEG-4 Requirement", Mar. 1994.

[64] F. Peraira, "MPEG4: a New Challenge for the Representation of Audio-Visual Information", *Proc. International Picture Coding Symposium (PCS)*, vol. 1, pp. 7-16, Melbourne, Australia, Mar. 1996.

[65] MSDL Ad Hoc Group, "Requirements for the MPEG4 Syntactic Description Language", Doc. ISO/IEC JTC1/SC29/WG11 N1022, Jul. 1995.

[66] F. Peraira (editor), "MPEG4 Testing and Evaluation Procedures Document", Doc. ISO/IEC/JTC1/SC29/WG11 N999, Tokyo meeting, Jul. 1995.

[67] MPEG Video Group, "MPEG4 Video Verification Model 1.0", Doc. ISO/IEC/JTC1/SC29/WG11 N1172", Munich meeting, Jan. 1996.

[68] T. Chiang and D. Anastassiou, "Hierarchical Coding of Digital Television", *IEEE Communications Magazine*, pp. 38-45, May. 1994.

[69] A. Sanz, C. Munoz and N. Garcia, "Approximation Quality Improvement Techniques in Progressive Image Transmission", *IEEE Journal on Selected Areas in Communications*, vol. SAC-2, pp. 339-373, Mar. 1984.

[70] M. I. Szean, M. Rabbani and P. W. Jones, "Progressive Transmission of Images using a Prediction/Residual Encoding Approach", *Optical Engineering*, vol. 28, no. 5, pp. 556-564, 1989.

[71] K. H. Tzou, "Progressive Image Transmission: A Review and Comparison of Techniques", *Optical Engineering*, vol. 26, no. 7, pp. 581-589, July 1987.

[72] J. Naor and S. Peleg, "Hierarchical Image Representation for Compression, Filtering and Normalization", *Pattern Recognition Letters*, pp. 43-46, Oct. 1983.

[73] K. N. Ngan, "Image Display Techniques using the Cosine Transform", *IEEE Transaction on ASSP*, vol. LASSP-32, pp. 173-177, Feb. 1984.

[74] K. H. Tzou and S. E. Elnabas, "An Optimum Progressive Transmission and Reconstruction Scheme for Transformed Image", *Proc. IEEE Int. Conf. On Communications*, Toronto, Ont., pp. 413-418, Jun. 1986.

[75] J. M. Beaumont, "Image Data Compression using Fractal Techniques", *British Telecom Technological Journal*, vol. 9(4), pp. 93-108. 1991.

[76] E. Walch and E. Karnin, "A Fractal Based Approach to Image Compression," *Proc. of ICASSP*, pp 529-532, 1986.

[77] E. W. Jacobs, Y. Fisher and R. D. Boss, "Image Compression: A Study of the Iterated Transform Method", *Signal Processing*, vol. 29, pp. 25-26, 1992.

[78] A. E. Jacquin, "Fractal Image Coding Based on a Theory of Iterated Contractive Image Transformations", *Visual Communications and Image Processing*, vol. SPIE-1360, 1990.

[79] D. M. Monro, "A Hybrid Fractal Transform", *Proc. ICASSP 93*, pp. v: 169-72.

[80] A. E. Jacquin, "Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformaitons", *IEEE Trans on Image Processing*, vol. 1, pp. 18-30, Jan. 1992.

[81] M. F. Barnsley, A. E. Jacquin, "Application of Recurrent Iterated Function Systems to Images", *Visual Communications and Image Processing*, vol. SPIE-1001, 1988.

[82] I. Corset, S. Jeannin, L. Bouchard, "MPEG-4: Very Low Bit Rate Coding for Multimedia Applications", *Proc. SPIE*, vol. 2308, pp. 1065-1073, 1994.

[83] D. Anastassiou, "Current Status of the MPEG-4 Standardization Effort", *Proc. SPIE*, vol. 2308, pp. 16-24, 1994.

[84] S. Panchanathan, A. Jain and N. Gamaz, "Scalable Image Compression Using Combined Wavelet Transform and Vector Quantization", *Proc. SPIE*, vol. 2419, pp. 354-364, 1995.

[85] Q. Hu and S. Panchanathan, "Spatial Scalability in Compressed Domain", *Proc. SPIE Symposium on Electronic Imaging Science and Technology - Digital Video Compression-Algorithms and Technologies*, vol. 2668, pp. 60-68, San Jose, California, USA, Feb. 1996.

[86] Q. Hu and S. Panchanathan, "Spatial Scalability in JPEG Compressed Domain", *Proc. Picture Coding Symposium (PCS)'96*, vol. 1, pp. 29-33, Melbourne, Australia, Mar. 1996.

[87] Q. Hu and S. Panchanathan, "A Comparative Evaluation of Spatial Scalability Techniques in the Compressed Domain", *Proc. Canadian Conference on Electrical and Computer Engineering (CCECE'96)*, vol.1, pp. 474-477, Calgary, Canada, May, 1996.

[88] Q. Hu and S. Panchanathan, "Image/Video Spatial Scalability in Compressed Domain", to appear in the special issue on *Multimedia Communications in the IEEE Trans. on Industrial Electronics*, Apr. 1997.

[89] Q. Hu and S. Panchanathan, "Encoding Scaled MPEG Video in Compressed Domain", to appear in the *Proc. Visual Communications and Image Processing '97*, San Jose, California, USA, Feb. 1997.