Scalable 3-D wavelet video coding

Zong, Wenbo

2008

Zong, W. (2008). Scalable 3-D wavelet video coding. Master's thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/13120>

<https://doi.org/10.32657/10356/13120>

SCALABLE 3-D WAVELET VIDEO CODING

ZONG WENBO

School of Electrical and Electronic Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Master of Engineering

2008

Acknowledgements

First and foremost, I would like to thank my supervisor, Prof. Kai-Kuang Ma, for his professional guidance, invaluable assistance, patience, and kindly encouragement throughout the two years' study as well as his constructive comments on this thesis.

It is also my great pleasure to thank my labmates in Multimedia Technology Lab at the School of Electrical and Electronic Engineering, Nanyang Technological University. Discussing with them has been a pleasant and fruitful experience.

I am grateful to my colleagues at STMicroelectronics. Though not directly related to my study, working with them has nevertheless strengthened my understanding on signal processing and broadened the horizon of my knowledge.

I would also like to thank the reviewers for their valuable suggestions that help improve the quality of this thesis. Some of the questions are thought-provoking and searching for the answers deepens my understanding on the topic.

Lastly but most importantly, this thesis is dedicated to my family, whose love and support are always the greatest inspiration for me in my life.

Contents

Acknowledgements	i
List of Figures	v
List of Tables	viii
List of Acronyms	x
Abstract	xiii
1 Introduction	1
1.1 Motivation	1
1.2 Scalability in Existing Hybrid Coders	2
1.3 The New 3-D Wavelet Coding Approach	7
1.4 Thesis Outline	9
2 Basics of Filter Banks and Wavelets	10
2.1 Series Expansion of Signals	10
2.2 Filter Banks	12
2.2.1 Orthogonal Filter Bank	13
2.2.2 Biorthogonal Filter Bank	15
2.3 Wavelets and Wavelet Transform	16
2.3.1 Computation of Wavelet Transform Using Filter Banks	20
2.3.2 Multi-Dimensional Wavelet Transform	22
2.4 Lifting Factorization of Wavelet Transform	22
3 Motion-compensated Wavelet Coding (MCWC)	25
3.1 Overview of Motion-compensated Wavelet Coding	25
3.2 Motion-compensated Temporal Filtering (MCTF)	28
3.2.1 How MCTF Operates	29
3.2.2 MCTF with Lifting Haar Transform	33
3.2.3 MCTF with Lifting 5/3 Transform	34
3.2.4 Discussion	36

3.3	Three MCWC Schemes and Their Scalability Issues	37
3.3.1	The t+2D Scheme	38
3.3.2	The 2D+t Scheme	41
3.3.3	The 2D+t+2D Scheme	45
3.3.4	Discussion	46
4	Spatial Scalability in the t+2D Scheme	48
4.1	Problems with Scaled Motion Field	48
4.1.1	Phase Mismatch	51
4.1.2	Improved Decoding with Up-scaling	53
4.1.3	Experimental Results and Discussion	54
4.2	Anatomy of the MCTF Process	55
4.2.1	Subband Scrambling	60
4.3	Encoder Side Solution	63
4.3.1	Fundamental Difference between t+2D and 2D+t with Complete-to-Overcomplete DWT (CODWT)	65
4.3.2	Experimental Results and Discussion	65
4.4	One Practical Solution: The Pyramidal t+2D Scheme	71
4.4.1	Experimental Results and Discussion	72
5	Set-partitioning in MCWC	76
5.1	Variable Size Block Motion Compensation	76
5.2	Set-partitioning According to Motion Blocks	81
5.3	Experiments	83
5.3.1	Experiment 1	83
5.3.2	Experiment 2	85
5.3.3	Discussion	87
6	Concluding Remarks and Future Work	89
6.1	Contributions of the Thesis	89
6.2	Future Research Directions	90
A	Review of Embedded Wavelet Image Coding	91
A.1	The Embedding Principle	91
A.2	Entropy Rate	92
A.3	Overview of Embedded Wavelet Image Coding	93
A.3.1	Wavelet Transform	94
A.3.2	Bitplane Coding	96
A.3.3	Set-partitioning	99
A.4	Zerotree Coding	99
A.5	Zero-block Coding	101
A.6	Morphological Coding	102

A.7 Discussion	103
B Experiment Results: Entropy of Foreman (GOP 1)	105
C Experiment Results: Entropy of Mobile (GOP 1)	114
Bibliography	123

List of Figures

1.1	Hybrid video coding system	2
1.2	Generic layered coding structure of the hybrid coding	3
1.3	Dyadic hierarchical structure for three temporal layers	4
1.4	Laplacian pyramid for spatial scalability	5
1.5	SNR scalable structure with two layers in hybrid coders	6
1.6	MPEG-2 scalable decoder	6
1.7	MPEG-4 FGS decoder	7
2.1	Two-band analysis and synthesis filter banks	12
2.2	Sine wave and wavelet	17
2.3	Filter bank implementation of forward and inverse orthogonal wavelet transforms.	20
2.4	Two-band filter bank iteration tree.	21
2.5	Frequency bands for the analysis tree	21
2.6	2D wavelet transform with 1D wavelet filters.	22
2.7	Illustration of directionality of 2D wavelets	23
2.8	The lifting wavelet transform	24
3.1	The general 2D+t+2D framework	26
3.2	The spatiotemporal wavelet decomposition tree	27
3.3	Temporally filtered frames with and without motion compensation	30
3.4	Motion connection classification and corresponding temporal filtering	31
3.5	MCTF with lifting Haar transfrom	34
3.6	MCTF with lifting biorthogonal 5/3 transfrom	35
3.7	The t+2D wavelet coding system	39
3.8	Encoding and decoding steps for lower spatial resolution video in t+2D	40
3.9	The PSNRs resulted from spatially-scaled decoding in t+2D	41
3.10	The 2D+t wavelet coding system	42
3.11	Encoding and decoding steps for lower spatial resolution video in 2D+t	43
3.12	Formation of the prediction reference for the HL band with CODWT	45
3.13	The overall 2D+t+2D encoding architecture	46
4.1	Lifting Haar operations	49

4.2	The generalized lifting Haar operation	49
4.3	Ideal approach to producing C_{SL}	50
4.4	Conventional approach to producing C_{SL}^c	52
4.5	Proposed approach to producing C_{SL}	53
4.6	PSNR performance of proposed decoder vs. conventional decoding. CIF sequences are decoded at QCIF resolution and 30 fps.	56
4.7	PSNR performance of proposed decoder vs. conventional decoding. CIF sequences are decoded at QQCIF resolution and 30 fps.	57
4.8	The detailed breakdown of the lifting step	59
4.9	Subband leakage illustration: wavelet data of spatial low-pass frames before and after MCTF.	61
4.10	Subband leakage illustration: wavelet data of spatial high-pass frames before and after MCTF.	62
4.11	Detailed steps to generate the prediction/update signal of the proposed LTH-MCTF approach.	64
4.12	The 2D+t scheme with CODWT where spatial scales are self-contained.	66
4.13	The 2D+t scheme with CODWT where spatial scales are not self-contained.	67
4.14	PSNR performance of the proposed LTH-MCTF encoder for full resolution.	68
4.15	PSNR performance of the proposed LTH-MCTF encoder with and without proposed decoder for half resolution.	69
4.16	PSNR performance of the proposed LTH-MCTF encoder with and without proposed decoder for quarter resolution.	70
4.17	The proposed pyramidal t+2D scheme	72
4.18	PSNR performance of the proposed pyramidal t+2D scheme for half resolution.	74
4.19	PSNR performance of the proposed pyramidal t+2D scheme for quarter resolution.	75
5.1	Example of motion block partitioning in hierarchical motion estimation	78
5.2	Illustration of the interaction between R_{mv} and R_{3D} with respect to the total distortion.	79
5.3	The rate-distortion curve.	80
5.4	Illustration of motion estimation block size and residues	82
5.5	Naming conventions for spatial and temporal subbands	83
5.6	Average reduction in entropy (of significance information) with set-partitioning	86
A.1	Illustration of the embedding principle.	92
A.2	Embedded wavelet image coding system	94
A.3	Wavelet filter spectrum	94
A.4	Separable wavelet decomposition	95
A.5	Subbands after three-level dyadic decomposition	95
A.6	The original Lena image (512x512)	97
A.7	Wavelet coefficients of the Lena image (512x512)	97
A.8	Bitplane coding illustration	98
A.9	Rate-distortion curve of bitplane coding	98

A.10 Parent-children relationship in a spatial orientation tree	100
A.11 Illustration of the decaying spectra of images	101
A.12 Quadtree construction and traversing	102
A.13 Illustration of dilation-run coding	103
A.14 Probability distribution functions affect compressibility	104

List of Tables

3.1	Comparison of three schemes of MCWC	47
4.1	List of notations	50
5.1	Parameter settings for MC-EZBC	84
5.2	PSNR (dB) comparison with vs. without set-partitioning according to motion blocks. LIP context models modified.	87
5.3	PSNR (dB) comparison with vs. without set-partitioning according to motion blocks. LIN context models modified.	87
B.1	Foreman Subband 0	106
B.2	Foreman Subband 1	106
B.3	Foreman Subband 2	107
B.4	Foreman Subband 3	107
B.5	Foreman Subband 4	108
B.6	Foreman Subband 5	108
B.7	Foreman Subband 6	109
B.8	Foreman Subband 7	109
B.9	Foreman Subband 8	110
B.10	Foreman Subband 9	110
B.11	Foreman Subband 10	111
B.12	Foreman Subband 11	111
B.13	Foreman Subband 12	112
B.14	Foreman Subband 13	112
B.15	Foreman Subband 14	113
B.16	Foreman Subband 15	113
C.1	Mobile Subband 0	115
C.2	Mobile Subband 1	115
C.3	Mobile Subband 2	116
C.4	Mobile Subband 3	116
C.5	Mobile Subband 4	117

C.6 Mobile Subband 5	117
C.7 Mobile Subband 6	118
C.8 Mobile Subband 7	118
C.9 Mobile Subband 8	119
C.10 Mobile Subband 9	119
C.11 Mobile Subband 10	120
C.12 Mobile Subband 11	120
C.13 Mobile Subband 12	121
C.14 Mobile Subband 13	121
C.15 Mobile Subband 14	122
C.16 Mobile Subband 15	122

List of Acronyms

3-D ESCOT Three-Dimensional Embedded Subband Coding with Optimal Truncation

BP Bit Plane

CIF Common Intermediate Format

CODWT Complete-to-Overcomplete Discrete Wavelet Transform

DCT Discrete Cosine Transform

DPCM Differential Pulse Code Modulation

DWT Discrete Wavelet Transform

EWIC Embedded Wavelet Image Coding

EBCOT Embedded Block Coding with Optimal Truncation

EZBC Embedded ZeroBlock Coding and context modeling

EZW Embedded Zerotree Wavelet

FB Filter Bank

FGS Fine Granularity Scalability

FIR Finite Impulse Response

FPS Frame Per Second

GOP Group of Pictures

HD High Definition

HVSBM Hierarchical Variable Size Block Matching

IDCT Inverse Discrete Cosine Transform

IDWT Inverse Discrete Wavelet Transform

IIR Infinite Impulse Response

IMCTF Inverse Motion-Compensated Temporal Filtering

JPEG Joint Photographic Experts Group

LBS Low-Band Shift

LSB Least Significant Bit

LSI Linear and Shift-Invariant

LSV Linear and Shift-Variant

MC Motion Compensation/Compensated

MCP Motion-Compensated Prediction

MCTF Motion-Compensated Temporal Filtering

MCWC Motion-Compensated Wavelet Coding

ME Motion Estimation

MSB Most Significant Bit

MSE Mean Squared Error

MPEG Moving Picture Experts Group

MV Motion Vector

MVF Motion Vector Field

PCM Pulse Code Modulation

PDF Probability Density Function

PR Perfect Reconstruction

PRFB Perfect Reconstruction Filter Bank

PSNR Peak Signal-to-Noise Ratio

PMF Probability Mass Function

R-D Rate-Distortion

QCIF Quarter Common Intermediate Format

SAD Sum of Absolute Difference

SBC Subband Coding

SFQ Space-Frequency Quantization

SH Spatial High-pass

SL Spatial Low-pass

SNR Signal-to-Noise Ratio

SPIHT Set-Partitioning In Hierarchical Trees

VSBE/VSBMC Hierarchical Variable Size Block Motion Estimation/Compensation

Abstract

Concerning the various applications that involve heterogeneous networks, computing capabilities, and/or display terminals, such as Internet video streaming and live video surveillance, a scalable video coding system provides an effective solution to address this issue by offering some scalable functionalities including temporal scalability, spatial scalability and signal-to-noise ratio (SNR) scalability, while achieving high compression performance. Conventional hybrid coders like MPEG-2 and MPEG-4 are not able to offer scalability efficiently due to the recursive motion-compensated (MC) prediction loop. In recent years, *motion-compensated wavelet coding* (MCWC) has emerged as a promising video coding approach to offering both high compression performance and versatile scalabilities.

In this research, we first introduce the 3-D MCWC technology and its three possible structures: $t+2D$, $2D+t$, and $2D+t+2D$. We then investigate two issues related to MCWC. The first issue is the spatial scalability of the $t+2D$ scheme, which suffers from degraded rate-distortion (R-D) performance for lower spatial resolutions. We investigate the problem from both the decoder side and the encoder side. At the decoder side, we present an in-depth analysis on the inverse *motion-compensated temporal filtering* (MCTF), or IMCTF, when reconstructing lower spatial resolution videos, and propose a method to improve the IMCTF process at the decoder. At the encoder side, we analyze the MCTF process in detail, and highlight the root source of aliasing that is not cancellable for lower spatial resolutions at the decoder. We propose a scheme named *low-to-high* lifting MCTF (LTH-MCTF) to eliminate this aliasing. Furthermore, we propose a *practical* solution, named *pyramidal* $t+2D$, that applies $t+2D$ coding to each level of the frame pyramids to produce one separate bitstream with optimal R-D performance for each spatial resolution.

The second issue we investigate in this research is the possibility of applying set-partitioning techniques to the temporal high-pass frames generated by variable size block motion compensation. We observe that the MC prediction residues exhibit different statistics in motion blocks of different sizes. We thus propose to partition the wavelet coefficients into groups according to the size of the motion block to which they correspond, which is shown to reduce the entropy with bitplane quantization.

Chapter 1

Introduction

1.1 Motivation

With the advances in networking technology and chips processing power, digital video is becoming more and more accessible, evidenced by the mobile video service on portable devices, digital TV, video over IP, to name a few. Obviously, these services demand different qualities and resolutions of video contents. For example, we would not expect high-definition video quality being delivered on the mobile phone that we would do otherwise on a digital high-definition (HD) TV, and vice versa. This shows that the requirements for different applications are very often heterogeneous. However, these heterogeneous requirements may not be foreseeable at the time when the digital video bitstreams were generated. Even for a specific visual communication application, the requirements may not be completely determined in advance. For instance, consider video over IP that is enjoying a growing popularity today, the encoder might need to generate a bitstream optimized for a range of bitrates rather than at a fixed bitrate, since the channel bandwidth and error conditions vary with time and different networking environments. Hence, instead of encoding the video at a targeted quality and resolution for a specific application every time, it would be more attractive to have an *encode-once-and-decode-multiple-times* coder such that an embedded scalable bitstream is created at the highest video quality, resolution and frame rate so that its partial bitstreams can also be decoded at a lower quality, resolution and/or frame rate according to the requirements (e.g., channel capacity, storage capacity, processing power, display size). This kind of flexibility of partially decoding an encoded bitstream to meet the heterogeneous requirements of various applications is achieved in a limited way in conventional hybrid coders, such as MPEG-2 and MPEG-4. In the following section, we shall briefly review the issues of scalability encountered in the conventional hybrid coders, especially on their SNR (quality) scalability. This would clearly establish concrete justification for the necessity of investigating and designing a scalable coder.

(a) Encoder structure

(b) Decoder structure

Figure 1.1: Hybrid video coding system. T : transform, Q : quantization, T^{-1} : inverse transform, Q^{-1} : inverse quantization.

1.2 Scalability in Existing Hybrid Coders

Current video coding standards, e.g., MPEG-2 [1], MPEG-4 [2], and H.264 [3], are all based on the so-called *hybrid coding* scheme, consisting of *intraframe* coding and *interframe* coding. In the intraframe coding, transform coding is employed to compact the energy in the spatial domain to only a few significant transform coefficients for better compression. In the interframe coding, motion compensation is used to remove the temporal redundancy. The structure of a typical hybrid coding system is shown in Fig. 1.1. Note that the encoder mimics its decoder's behavior by running a local decoding loop, so that the prediction of the next frame is constructed based on the *decoded* current frame instead of the original current frame that will be not available at the decoder. This prediction loop, with motion compensation, is recursive in nature, meaning that the prediction for the next frame is dependent on the *decoded* current frame, which in turn depends on the previously *decoded* frame, and so on.

Temporal, *spatial*, and *SNR* scalabilities in such a hybrid coder are achieved by a *layered* approach [4] including a *base* layer corresponding to the lowest quality of the original source and one or more *enhancement* layers containing complementary information for building up enhanced videos. The generic structure for scalable coding based on a layered approach is depicted in Fig. 1.2.

Figure 1.2: Generic layered coding structure of the hybrid coding that involves one base layer and a set of enhancement layers. The symbol $\downarrow$ denotes the down-scaling in the temporal or spatial dimension, while the symbol $\uparrow$ denotes the corresponding up-scaling operation in the same dimension.

The down-scaling operator (denoted by symbol $\downarrow$) and up-scaling operator (denoted by symbol $\uparrow$) are decimation and interpolation in the temporal dimension and/or the spatial dimension, where certain particular operations may be involved, e.g., spatial filtering to remove spectral distortion for spatial scalability, or motion-compensated processing for inserting frames for temporal scalability. The spatiotemporal resolution and quality of the base layer are dependent on the preceding down-scaling operation and the quantizer in its encoding stage, respectively. The reconstructed base layer serves as a starting base to the finer enhanced videos. For the spatial scalability case, it can be up-scaled to the first enhancement layer's resolution. Likewise, this reconstructed enhancement video can be further up-scaled to the next enhancement layer's resolution to serve as a starting base to that enhanced video. The same concept applies to temporal and SNR scalability. It is easy to see that the finer enhancement layers are useless without the coarser layers. Furthermore, the base layer and the enhancement layers should be all self-contained, meaning that when a *scaled* video up to a certain level (in terms of SNR or resolution) is under construction it should not utilize any information from the finer enhancement layers. Should this rule not be observed, when reconstructing a scaled video the decoder will not utilize the same information as the encoder did, resulting in a *mismatch* between the decoder and the encoder.

Temporal scalability is an effective way to reduce the transmission bandwidth requirement. It is conceptually simple, as it mainly involves decimating the input sequence to form the base layer, and supplementing the skipped frames in the enhancement layers. There are three types of video frames: *I-frame*, *P-frame*, and *B-frame*. An *I-frame* is *intra-coded* and hence self-contained, and it is used as a reference frame to predict other frames. A *P-frame* is *inter-coded* with motion-compensated prediction from a previous frame that may be an *I-frame* or a *P-frame*, and it can also be used as a reference frame to predict other frames. A *B-frame* is also *inter-coded* using motion-compensated prediction, but the prediction is done bidirectionally from both a previous frame and a future frame.

Figure 1.3: Dyadic hierarchical structure for three temporal layers. Black-shaded frames labelled with I_0/P_0 (meaning I -frame and P -frame, respectively), green-shaded frames labelled with B_1 and blue-shaded frames labelled with B_2 belong to the base layer, the first enhancement layer and the second enhancement layer, respectively. The solid arrows indicate the directions of motion-compensated prediction performed. The dashed arrows in the base layer indicate optional MC prediction, applicable only when the frame is viewed as P -frame.

In principle, one layer can contain frames of any type, as long as the reference frames being used to generate the prediction signals are from the same layer or coarser enhancement layers. Temporal scalability does not necessarily increase the overall frame rate, as the frames already included in the coarser layers do not need to be coded again. In fact, they can be even further used to predict the frames in the current layer. Hence, in this case it is more appropriate to interpret the down-scaling and the up-scaling operation in Fig. 1.2 as motion-compensated prediction for the decimated frames instead of spatial scaling. Furthermore, temporal scalability based on the B -frames only does not necessarily lead to a loss in performance, as the coder may essentially perform the same way as a single-layer coder. For example, the dyadic hierarchical structure for temporal scalability shown in Fig. 1.3 can be realized by the flexible frame prediction mode of H.264 [3].

Spatial scalability is typically realized by the generic structure as shown in Fig. 1.4, which consists of the *Gaussian pyramid* and the *Laplacian pyramid* [5]. In general, there are two cases for the coarse-to-fine spatial prediction in scalable layered video coding. First, the Gaussian (approximation) pyramid is generated based on the down-scaling operation. The Laplacian (prediction residual) pyramid is formed by conducting inter-layer prediction from the coarser layer to the next finer layer. The residual frames in each layer are then coded either using an image coding technique or applying a single-layer video coding method with motion-compensated prediction at each layer. Second, the Gaussian pyramid is generated only for the I -frames, and the corresponding Laplacian pyramid from these frames is generated by inter-layer prediction. Each layer is then coded using motion-compensated prediction as in the case of single-layer video coding, except that recon-

Figure 1.4: Laplacian pyramid for spatial scalability. The dark-gray-shaded boxes represent approximation signals, and the light-gray-shaded boxes represent prediction residues.

structuring *I-frames* utilizes the prediction information from the previous coarser layer. Due to the over-complete nature of the Laplacian pyramid, the overall coding performance for the full resolution is inferior to that of single-layer video coder.

SNR scalability provides a mechanism to gradually build up visual quality through a set of layers where the quantizer step size decreases from the coarser layers to the finer layers (see Fig. 1.5). The base layer is coded in the same way as a non-scalable coder. Each enhancement layer encodes the difference between the original input and the output of the previous layer. Therefore, the base layer corresponds to the minimum bitrate, and all layers together utilize the maximum bitrate. All factors being equal, the bitrate of the base layer depends on the quantizer step size and the energy of MC prediction residues. In other words, to achieve a certain bitrate for the base layer, we have to either minimize the energy of the MC prediction residues and accordingly use a small quantizer step size, or tolerate higher energy for the MC prediction residues by using a larger quantizer step size. The advantage of the former case compared to the latter case is two-fold. First, the base layer output will have less distortion. Second, the input to the next enhancement layer will have lower energy and hence requires lower bitrate for that enhancement layer, which will lead to better overall coding performance. To minimize the energy of MC prediction residues in the base layer, the enhancement layer(s) has to be used to compose a prediction reference frame that is closer to the input frame. This is exactly what the MPEG-2 standard does for SNR scalability, though the standard itself only stipulates the SNR scalable decoder as shown in Fig. 1.6. It is easy to see that, when decoding a *scaled* video with some fine enhancement layer(s) dropped, the decoder will not see the same input as the encoder did, causing a mismatch between the decoder and the encoder. The problem worsens when the prediction loop involves a few frames, because errors incurred in reconstructing one frame will be propagated to its subsequent frames, causing a *drift problem*. It has been shown that the

Figure 1.5: SNR scalable structure with two layers in hybrid coders. The processing in the dashed box is optional, meaning that motion-compensated prediction may or may not be carried out for the enhancement layer.

SNR performance degrades quickly with time due to the drift problem [6].

Concerning the drift problem caused by using finer enhancement layer(s) for prediction of the base layer, the alternative is to exclude the finer enhancement layer(s) in the MC prediction loop, making the base layer self-contained. This will result in the opposite situation of what has been just described. For the base layer, the energy of the MC prediction residues will be higher and hence the distortion will be higher for a fixed bitrate (which is the minimum bitrate in the bitrate range). Consequently, The input to the next enhancement layer will have higher energy and hence requires higher bitrate for that enhancement layer. The overall coding efficiency is thus reduced. This *drift-free* approach is employed by the *fine granularity scalability* (FGS) in MPEG-4. The FGS decoder is depicted in Fig. 1.7. In FGS, the base layer operates at the lower bound of the bitrate range, and

Figure 1.6: MPEG-2 scalable decoder [1]

Figure 1.7: MPEG-4 FGS decoder [2]

the enhancement layer is bitplane coded (see Section A.3.2) so that it provides progressive quality improvement to the base video. It has been shown that there is a loss of 2 dB at the upper bound of the bitrate range compared to non-scalable coding [7].

Remark: As can be seen from the above description, for SNR scalability we have a basic dilemma of penalizing either the enhancement layer as in MPEG-4 FGS, or the base layer as in MPEG-2. The existence of the recursive prediction loop makes hybrid coders inherently limited in SNR scalability. Sophisticated techniques are needed to control the drift, which would increase the already high complexity of hybrid encoders. Simultaneous temporal, spatial and SNR scalability is therefore very difficult to achieve in such recursive prediction-based coders.

1.3 The New 3-D Wavelet Coding Approach

The success of embedded and scalable wavelet coding for images, e.g., EZW [8], SPIHT [9], EZBC [10] and JPEG2000 [11, 12, 13], promises a potential success for video coding. In light of the inefficiency of conventional hybrid coders in scalability, it might be better off simply abandoning the prediction structure and employing a three dimensional (3-D) wavelet/subband coding scheme that is scalable in nature. The *motion-compensated wavelet coding* (MCWC) [14, 4, 15, 16, 17, 18, 19] is precisely the result of such a major paradigm shift from the conventional hybrid coding. Hence, MCWC is in essence a 3-D subband coding approach. It has three major features:

- **Wavelet transform:** The video signal requires to be decomposed into 3-D spatiotemporal subbands through wavelet transform. The decomposition may take place first in the temporal dimension and then in the spatial dimension, which is referred to as a t+2D scheme. Alternatively, the decomposition may take place in the reverse order, which is referred to as a 2D+t scheme. Yet, there exists a third scheme, called 2D+t+2D, that is most general. Due to the inherent multi-resolution property of wavelets, a properly chosen subset of the 3-D subbands corresponds to a particular frame rate and/or spatial resolution.

- **Embedded wavelet coding:** The spatiotemporal subbands resulted from 3-D decomposition need to be encoded in such a way that an embedded bitstream can be produced. Well-established *embedded wavelet image coding* (EWIC) techniques can be directly employed to code each 2-D subband separately, or they can be extended to include the temporal dimension to code the 3-D subbands jointly. The embedded nature of the coded bitstream can also enable better protection for the more important data when transmitting over a noisy channel.
- **Motion-compensated temporal filtering (MCTF):** When wavelet transform is carried out in the temporal domain, the video frames are filtered into a low-pass subband, that corresponds to a video of half frame rate, and a high-pass subband. Since video objects may move in any direction, directly applying wavelet transform to pixels at the same positions of different frames tends to yield blurred low-pass frames and distribute significant energy into the high-pass frames. This will lead to a loss in coding efficiency. Hence, wavelet transform is applied along the motion paths to better exploit the temporal correlations, which is equivalent to performing motion compensation (MC). This form of temporal decomposition or filtering is referred to as *motion-compensated temporal filtering* (MCTF) in the literature. Lifting wavelet transform is often employed in MCTF, because it allows subpixel-accurate motion vectors (MV) for more efficient MC while still achieving perfect reconstruction (PR).

Our research work has been focusing on the emerging MCWC technology, for which a brief review is given. We investigate the spatial scalability of the t+2D scheme, which suffers degraded PSNR performance for the lower spatial resolutions. This issue is examined from both the decoder side and the encoder side. At the decoder, we will reveal the sub-optimality of the conventional approach to reconstructing a lower spatial resolution, by presenting an in-depth analysis on the inverse MCTF (IMCTF) process. A simple method is proposed to improve the decoder. However, the improved decoder does not solve the whole problem because some amount of aliasing is already incurred in the lower spatial subbands at the encoder. Therefore, we analyze in detail the MCTF process at the encoder, and highlight the root source of aliasing that is not cancellable by the decoder for lower spatial resolutions. We propose a *low-to-high* lifting MCTF (LTH-MCTF) scheme to eliminate this aliasing, which greatly improves the PSNR performance for lower spatial resolutions but degrades the performance for the full resolution. This is in contrast to the original t+2D scheme that gives the best PSNR performance for the full spatial resolution, but shows degrading performance for lower spatial resolutions. Hence, we also propose a *practical* solution, named *pyramidal t+2D*, that applies t+2D coding to each level of the frame pyramids to produce one separate bitstream with optimal performance for each spatial resolution.

In this research, we also investigate the possibility of applying set-partitioning techniques to the temporal high-pass frames generated by variable size block motion compensation. The motivation behind is that image coding methods used in MCWC are often very sophisticated and designed for natural images but the high-pass frames in general do not resemble natural images. Hence, simple image coding techniques employing set-partitioning strategies may perform comparably, with the added advantage of lower cost. We will show that when variable size block motion compensation is used, the MC prediction residues exhibit different statistics in motion blocks of different sizes.

1.4 Thesis Outline

The remainder of this thesis is organized as follows:

- **Chapter 2** presents a brief review on the basics of wavelet transform and filter banks, and the lifting implementation. The material provided in this chapter is intended to facilitate the discussion involving wavelet transform in the later chapters.
- **Chapter 3** introduces the emerging *motion-compensated wavelet coding* paradigm and describes in detail its core functional component, *motion-compensated temporal filtering* (MCTF). The three major schemes that an MCWC coder may assume are introduced, from which their differences and relative advantages and disadvantages are highlighted.
- **Chapter 4** presents an in-depth analysis of the spatial scalability problem incurred in the t+2D scheme. Both the IMCTF process at the decoder and the MCTF process at the encoder are analyzed in detail, giving insight into the root cause of the degraded PSNR performance for lower spatial resolutions. Solutions at the decoder side and the encoder side are proposed. As a by-product, the fundamental difference between the t+2D and 2D+t schemes can be observed and presented in this chapter. A practical solution is also proposed by considering the real world constraints such as transmission cost, storage cost, and computational cost.
- **Chapter 5** studies the statistical properties of the temporally filtered high-pass frames. A set-partitioning scheme is employed to take advantage of the different statistics in motion blocks of different sizes. Experiments show that a reduction in entropy has been achieved, however the final PSNR performance is not improved. Extensive experimental results are documented in Appendix B and Appendix C.
- **Chapter 6** concludes this thesis and provides some pointers to future research.
- **Appendix A** presents a brief introduction to EWIC, as the spatiotemporal subbands after 3-D decomposition in MCWC are often coded by some EWIC method. This will make the thesis more self-contained and readable.
- **Appendix B** documents the results of the entropy reduction experiments conducted in Chapter 5 for the *Foreman* test sequence.
- **Appendix C** documents the results of the entropy reduction experiments conducted in Chapter 5 for the *Mobile and Calendar* test sequence.

Chapter 2

Basics of Filter Banks and Wavelets

In this chapter, a brief introduction to the basics of filter banks and wavelets is presented as the fundamental background to serve the follow-up development. Filter banks are an important tool often exploited in the field of signal processing to split a signal into a number of frequency bands (subbands) by passing the input signal through a bank of filters, so that each subband can be treated in a manner matching its (distinct) characteristics. Wavelets are a powerful tool for describing and representing functions or signals. The connection between wavelet transforms and filter banks have been established [20][21]; it turns out that wavelet transforms can be efficiently implemented by filter banks. In order to present the idea of wavelets and wavelet transforms, it is necessary to look at it from two perspectives: (1) series expansion and (2) signal processing. Thus, we will first introduce the basics of series expansion in Section 2.1, then describe the basic operations of (two-band) filter banks in Section 2.2. The scaling function and wavelet functions of wavelet systems are succinctly described in Section 2.3, where multi-resolution wavelet representation and multi-dimensional wavelet transforms are also discussed. As the last topic in this chapter, Section 2.4 introduces the *lifting* implementation of wavelets, known as second generation wavelets, which is not only computationally more efficient but also invertible.

2.1 Series Expansion of Signals

A signal is a function of an independent variable, such as time or distance. In practice, a signal often can not be expressed in closed mathematical form, hence dealing with it directly is difficult. It is thus desirable to represent or transform a signal in a form that is more convenient to describe, analyze, and process. This transformation does not provide any extra information about the signal; it still carries the same information but in a different form. In mathematics, very often we do this by expanding a signal or function into a series. For example, the well-known Fourier series expansion decomposes a signal into a weighted summation of sinusoids of different frequencies.

A signal or function can be expressed as a linear combination

$$f(t) = \sum_{i \in I} \alpha_i \varphi_i(t), \quad (2.1)$$

where I is the set of indices for the sum, α_i are the (real-valued) expansion coefficients, and $\{\varphi_i\}_{i \in I}$ are a set of (real-valued) functions of independent variable t . If the expansion is unique, or equivalently the elements of $\{\varphi_i\}_{i \in I}$ are linearly independent, the set $\{\varphi_i\}_{i \in I}$ forms a *basis* for the class of functions that can be so expressed; correspondingly, the collection of functions that can be expressed by such a basis is called a *function space*. A space that is particularly important in signal processing is $L^2(\mathbb{R})$. This is the space of all functions with a well-defined Lebesgue integral of the square of the modulus of the function whose independent variable is a number over the whole real line. The superscript ‘2’ indicates L^2 -norm, which is defined as

$$\|f\|_2 = \sqrt{\langle f, f \rangle}, \quad (2.2)$$

where $\langle \cdot, \cdot \rangle$ denotes the *inner product* of two functions, defined as

$$\langle f, g \rangle = \int_{-\infty}^{\infty} f(t)g^*(t)dt. \quad (2.3)$$

Of particular importance is the *orthonormal* bases. A basis $\{\varphi_i\}_{i \in I}$ is orthonormal if

$$\langle \varphi_i, \varphi_j \rangle = \delta(i - j). \quad (2.4)$$

That is, the basis functions are orthogonal to each other and are normalized to unity norm. The expansion coefficients $\{\alpha_i\}$ of a function f in an orthonormal basis $\{\varphi_i\}_{i \in I}$ are obtained by

$$\alpha_i = \langle \varphi_i, f \rangle. \quad (2.5)$$

Thus, f can be expressed as

$$f(t) = \sum_{i \in I} \alpha_i \varphi_i(t) = \sum_{i \in I} \langle f, \varphi_i \rangle \varphi_i(t). \quad (2.6)$$

One important property of the orthonormal basis is that it follows the Parseval’s theorem which says that the energy or norm of the vector $\boldsymbol{\alpha}$ formed by the expansion coefficients is equal to that of the original signal. That is,

$$\|f\|_2^2 = \|\boldsymbol{\alpha}\|_2^2. \quad (2.7)$$

Another important class of bases is the *biorthogonal* basis. For a biorthogonal basis, the expansion functions φ_i are not necessarily orthogonal to each other, but to a *dual basis set* $\{\tilde{\varphi}_i\}_{i \in I}$:

$$\langle \varphi_i, \tilde{\varphi}_j \rangle = \delta(i - j). \quad (2.8)$$

The expansion in (2.1) for a biorthogonal system gives

$$f(t) = \sum_{i \in I} \langle f, \tilde{\varphi}_i \rangle \varphi_i(t). \quad (2.9)$$

It can be seen that the orthogonal bases are a special case of biorthogonal bases when $\tilde{\varphi}_i(t) = \varphi_i(t), \forall i \in I$. Since the basis functions φ_i are not necessarily orthogonal to each other, a biorthogonal expansion in general does not follow the Parseval’s theorem.

Figure 2.1: Two-band analysis and synthesis filter banks

2.2 Filter Banks

Before introducing filter banks and the transfer functions involved, we first lay out the transfer functions of common downsampler and upsampler.

- The z-transform of the output Y after downsampling input X by a factor of D is given by

$$Y(z) = \frac{1}{D} \sum_{k=0}^{D-1} X(z^{\frac{1}{D}} e^{-j2\pi k/D}). \quad (2.10)$$

- The z-transform of the output Y after upsampling input X by a factor of U is given by

$$Y(z) = X(z^U). \quad (2.11)$$

A filter bank is a structure that decomposes a signal into a collection of sub-signals (subbands). Its usefulness lies in that these sub-signals may exhibit specific aspects of the original signal, which might be easier to deal with. A popular application of filter banks is subband coding of data, where coding scheme is devised to take advantage of specific characteristics of each sub-signal so that more efficient compression can be yielded.

Fig. 2.1 shows a two-band critically sampled filter bank system, consisting of one analysis bank and one synthesis bank. We consider the typical case where one subband corresponds to low-pass (LP) information and the other subband to high-pass (HP) information. We shall denote H_0 and G_0 as LP filters, H_1 and G_1 as HP filters. The input signal is passed through H_0 and H_1 separately and downsampled by a factor of 2, resulting in sub-signals $v_0(n)$ and $v_1(n)$, respectively. Note that $v_0(n)$ and $v_1(n)$ are each at half sample rate of $x(n)$ due to the critical downsampling. For synthesis, $v_0(n)$ and $v_1(n)$ are first upsampled by a factor of 2, then filtered with G_0 and G_1 respectively, before summing up to produce the final output. Very often, $v_0(n)$ and $v_1(n)$ are processed before synthesis; for example, they may be quantized to reduce bitrate as in a data compression application. In the absence of error/distortion introduced to the system, it is often desirable that the filter bank can provide a perfect reconstruction (PR); that is, $x'(n) = x(n)$. We shall take a closer look at how the PR requirement can be satisfied by properly choosing the analysis and/or synthesis filters.

At the analysis stage, signals $v_0(n)$ and $v_1(n)$ are obtained by downsampling u_0 and u_1 , respectively, where $U_0(z) = X(z)H_0(z)$ and $U_1(z) = X(z)H_1(z)$. Hence, from (2.10) we have

$$V_0(z) = \frac{1}{2}[X(z^{\frac{1}{2}})H_0(z^{\frac{1}{2}}) + X(-z^{\frac{1}{2}})H_0(-z^{\frac{1}{2}})], \quad (2.12)$$

$$V_1(z) = \frac{1}{2}[X(z^{\frac{1}{2}})H_1(z^{\frac{1}{2}}) + X(-z^{\frac{1}{2}})H_1(-z^{\frac{1}{2}})]. \quad (2.13)$$

At the synthesis stage, signals $w_0(n)$ and $w_1(n)$ are obtained by upsampling $v_0(n)$ and $v_1(n)$, respectively, where $W_0(z) = V_0(z^2)$ and $W_1(z) = V_1(z^2)$. Substituting (2.12) and (2.13), we have

$$W_0(z) = \frac{1}{2}[X(z)H_0(z) + X(-z)H_0(-z)], \quad (2.14)$$

$$W_1(z) = \frac{1}{2}[X(z)H_1(z) + X(-z)H_1(-z)]. \quad (2.15)$$

Hence, the synthesized output $X'(z)$ is given by

$$\begin{aligned} X'(z) &= W_0(z)G_0(z) + W_1(z)G_1(z) \\ &= \frac{1}{2}[X(z)H_0(z) + X(-z)H_0(-z)]G_0(z) \\ &\quad + \frac{1}{2}[X(z)H_1(z) + X(-z)H_1(-z)]G_1(z) \\ &= \frac{1}{2}[H_0(z)G_0(z) + H_1(z)G_1(z)]X(z) \\ &\quad + \frac{1}{2}[H_0(-z)G_0(z) + H_1(-z)G_1(z)]X(-z). \end{aligned} \quad (2.16)$$

In (2.16), the second half involving $X(-z)$ is the *aliasing* introduced by downsampling. For perfect reconstruction, $X'(z)$ should be a delayed and possibly magnitude-scaled version of $X(z)$, i.e., $X'(z) = cX(z)z^{-m}$, for some constants c and m . Clearly, PR can be achieved by imposing the following two conditions:

$$P(z) = H_0(z)G_0(z) + H_1(z)G_1(z) = cz^{-m}, \quad (2.17)$$

$$Q(z) = H_0(-z)G_0(z) + H_1(-z)G_1(z) = 0. \quad (2.18)$$

We shall consider two types of filter banks: *orthogonal filter bank* (OFB) and *biorthogonal filter bank* (BFB), and see how the above constraints can be satisfied.

2.2.1 Orthogonal Filter Bank

We consider the problem of constructing a two-band orthogonal filter bank [22][23] in this section. A two-band filter bank is orthogonal if the 2-translates of the analysis LP filter h_0 and analysis HP filter h_1 are orthogonal to each other; that is

$$\langle \mathbf{h}_0(n), \mathbf{h}_1(n - 2k) \rangle = 0, \quad \forall k \in \mathbb{Z}, \quad (2.19)$$

where the inner product $\langle \cdot, \cdot \rangle$ of two discrete functions h and g is defined as

$$\langle f, g \rangle = \sum_{n \in \mathbb{Z}} f(n)g^*(n). \quad (2.20)$$

We now present an approach to constructing a PR two-band filter bank. Assume the synthesis LP filter g_0 is orthogonal to its even shifts and its norm is 1, i.e.,

$$\langle \mathbf{g}_0(n), \mathbf{g}_0(n - 2k) \rangle = \delta(k), \quad (2.21)$$

$$\|g_0\|_2 = 1. \quad (2.22)$$

It implies that the length of g_0 must be even. We then choose the synthesis HP filter g_1 such that

$$G_1(z) = z^{-N+1}G_0(-z^{-1}), \quad (2.23)$$

which can be obtained by applying three operations in sequence:

- $z \rightarrow -z$: multiplies $g_0(n)$ with $(-1)^n$ to transform its low-pass spectrum into high-pass;
- $z \rightarrow z^{-1}$: time reverses $(-1)^n g_0(n)$;
- multiplication by z^{-N+1} : delays $(-1)^n g_0(-n)$ by $N - 1$ to make it causal.

Thus, the HP filter g_1 has impulse response given by

$$g_1(n) = (-1)^{N-1-n} g_0(N - 1 - n). \quad (2.24)$$

This special way of selecting g_1 , due to [24] and [25], ensures the following properties:

$$\langle \mathbf{g}_1(n), \mathbf{g}_1(n - 2k) \rangle = \delta(k), \quad (2.25)$$

$$\langle \mathbf{g}_0(n), \mathbf{g}_1(n - 2k) \rangle = 0. \quad (2.26)$$

That is, g_1 is orthogonal to its even shifts, and $\mathbf{g}_0(n - 2k)$ and $\mathbf{g}_1(n - 2l)$, $k, l \in \mathbb{Z}$, are mutually orthogonal.

It can be shown that the orthonormal set $\{\mathbf{g}_0(n - 2k), \mathbf{g}_1(n - 2l)\}_{k,l \in \mathbb{Z}}$ are an orthonormal basis for $L^2(\mathbb{Z})$ by verifying the Parseval's relation. Thus, if $f \in L^2(\mathbb{Z})$, it can be expanded as

$$f(n) = \sum_{k \in \mathbb{Z}} a_k g_0(n - 2k) + \sum_{k \in \mathbb{Z}} b_k g_1(n - 2k), \quad (2.27)$$

where

$$a_k = \langle \mathbf{f}(n), \mathbf{g}_0(n - 2k) \rangle, \quad (2.28)$$

$$b_k = \langle \mathbf{f}(n), \mathbf{g}_1(n - 2k) \rangle. \quad (2.29)$$

Note that the inner product is only computed with even shifts of g_0 and g_1 , which is equivalent to shifting the input f by two each time and performing the inner product with g_0 and g_1 , which in turn is equivalent to performing the inner product at each time instance and then downsampling by two. Such a series of inner products can be computed as convolving two signals with one signal time reversed, therefore the expansion coefficients $\{a_k\}$ and $\{b_k\}$ are computed by an analysis filter

bank with analysis filters h_0 and h_1 being the time-reversed version of g_0 and g_1 , respectively; that is,

$$H_0(z) = G_0(z^{-1}), \quad (2.30)$$

$$H_1(z) = G_1(z^{-1}). \quad (2.31)$$

Now we have constructed an orthonormal filter bank system starting with g_0 constrained by (2.25). It is easy to verify that the so-designed filter bank is perfectly reconstructible by substituting $H_0(z), H_1(z), G_0(z)$ and $G_1(z)$ into (2.17) and (2.18).

2.2.2 Biorthogonal Filter Bank

We now consider the problem of constructing a two-band biorthogonal filter bank [26][21]. With the same notations as established in Fig. 2.1, a biorthogonal filter bank system is such that

$$\langle \mathbf{h}_0(n), \mathbf{g}_0(n-2k) \rangle = \delta(k), \quad (2.32)$$

$$\langle \mathbf{h}_0(n), \mathbf{g}_1(n-2k) \rangle = 0, \quad (2.33)$$

$$\langle \mathbf{h}_1(n), \mathbf{g}_0(n-2k) \rangle = 0, \quad (2.34)$$

$$\langle \mathbf{h}_1(n), \mathbf{g}_1(n-2k) \rangle = \delta(k). \quad (2.35)$$

That is, the 2-translates of each of the analysis filters are orthogonal to all but one of the 2-translates of the synthesis filters.

In the biorthogonal case, instead of having g_0 orthogonal to its even shifts, we force it to be orthogonal to the even shifts of h_0 :

$$\langle \mathbf{h}_0(n), \mathbf{g}_0(n-2k) \rangle = \delta(k). \quad (2.36)$$

In order to satisfy the PR conditions, the high-pass filters h_1 and g_1 are chosen such that

$$h_1(n) = (-1)^n g_0(n+1), \quad (2.37)$$

$$g_1(n) = (-1)^n h_0(n-1), \quad (2.38)$$

for which the corresponding z-domain relationships are given by

$$H_1(z) = -zG_0(-z), \quad (2.39)$$

$$G_1(z) = -z^{-1}H_0(-z). \quad (2.40)$$

Note that when $h_0(n) = g_0(-n)$, we get the relationship in (2.24) for the orthogonal case.

We now verify that the PR conditions (2.17) and (2.18) hold for the filters so designed as above. Substitute (2.39) and (2.40) into (2.18), we obtain $Q(z) = 0$. Substitute into (2.17), we have

$$P(z) = H_0(z)G_0(z) + H_0(-z)G_0(-z), \quad (2.41)$$

whose time-domain relationship is given by

$$\begin{aligned}
p(n) &= h_0(n) * g_0(n) + ((-1)^n h_0(n)) * ((-1)^n g_0(n)) \\
&= (h_0 * g_0)(n) + (-1)^n (h_0 * g_0)(n),
\end{aligned} \tag{2.42}$$

where the operator $*$ denotes *convolution*.

Let us re-write the relationship expressed in (2.36) in an equivalent form as

$$(h_0 * g_0)(2k) = \delta(k). \tag{2.43}$$

Hence, for even n , (2.42) becomes

$$\begin{aligned}
p(n) \equiv p(2k) &= (h_0 * g_0)(2k) + (-1)^{2k} (h_0 * g_0)(2k) \\
&= 2(h_0 * g_0)(2k) \\
&= 2\delta(k),
\end{aligned} \tag{2.44}$$

and for odd n ,

$$\begin{aligned}
p(n) \equiv p(2k+1) &= (h_0 * g_0)(2k+1) + (-1)^{2k+1} (h_0 * g_0)(2k+1) \\
&= 0.
\end{aligned} \tag{2.45}$$

Obviously, the PR conditions are satisfied by the biorthogonal filter bank system as described above.

2.3 Wavelets and Wavelet Transform

A wavelet is a “short wave”, which has its (finite) energy concentrated in time and exhibits oscillating wave-like characteristic. It allows simultaneous time and frequency analysis and hence provides an important tool for a wide range of applications, such as functional analysis, signal processing, etc. It is in contrast to normal sinusoids that have equal magnitude over $-\infty < t < \infty$, as illustrated in Fig. 2.2.

A wavelet system has a set of fundamental *building blocks* used to construct or represent a signal or function. It yields a two-dimensional expansion set for some class of one-dimensional signals. For instance, if the wavelet set is given by $\{\varphi_{j,k}\}_{j,k \in \mathbb{Z}}$, then a linear expansion of a function f would be $f(t) = \sum_j \sum_k a_{j,k} \varphi_{j,k}(t)$, where $a_{j,k}$ are expansion coefficients. That is, wavelet expansion maps a one-dimensional signal (in variable t) into a two-dimensional array of coefficients (in variables j and k). It is this two-dimensional representation that allows localizing the signal in both time and frequency simultaneously, a property not possessed by Fourier series expansion.

All wavelet systems are generated from a single scaling function or wavelet by simple *scaling* and *translation*. We define a set of scaling functions in terms of integer translates of the basic scaling function by

$$\varphi_k(t) = \varphi(t - k), \quad k \in \mathbb{Z}, \quad \varphi \in L^2(\mathbb{R}), \tag{2.46}$$

Figure 2.2: Sine wave and wavelet

where $\mathbb{Z}$ is the set of all integers. The subspace of $L^2(\mathbb{R})$ spanned by these functions is defined as

$$\mathcal{V}_0 = \overline{\text{span}\{\varphi_k\}_{k \in \mathbb{Z}}}, \quad (2.47)$$

where

$$\text{span}\{\varphi_k\}_{k \in \mathbb{Z}} = \{\lambda_1 \varphi_1 + \lambda_2 \varphi_2 + \dots | \lambda_1, \lambda_2, \dots \in \mathbb{R}\} \quad (2.48)$$

and the over-bar denotes *closure* which means that a linear combination of members in the spanning set $\{\varphi_k\}_{k \in \mathbb{Z}}$ produces a member of the set. That is, if $f \in \mathcal{V}_0$, then there always exists a linear combination of φ_k that is equal to f :

$$f(t) = \sum_{k \in \mathbb{Z}} a_k \varphi_k(t) = \sum_{k \in \mathbb{Z}} a_k \varphi(t - k). \quad (2.49)$$

A family of functions with different time-frequency properties is generated by the two-dimensional parameterization of the basic scaling function by both scaling and translation:

$$\varphi_{j,k}(t) = 2^{j/2} \varphi(2^j t - k), \quad j, k \in \mathbb{Z}, \quad (2.50)$$

where the factor $2^{j/2}$ maintains a constant L^2 norm independent of scale j . At scale j , the span of $\varphi_{j,k}$ over k is

$$\mathcal{V}_j = \overline{\text{span}\{\varphi_{j,k}\}_{k \in \mathbb{Z}}} = \overline{\text{span}\{\varphi_k(2^j t)\}_{k \in \mathbb{Z}}}, \quad (2.51)$$

This means that if $f \in \mathcal{V}_j$,

$$f(t) = \sum_{k \in \mathbb{Z}} d_k \varphi_{j,k}(t) = \sum_{k \in \mathbb{Z}} d_k \varphi(2^j t - k), \quad (2.52)$$

where the factor $2^{j/2}$ has been absorbed into d_k .

It is easily seen that as j increases, the span becomes larger since $\varphi_{j,k}$ is narrower and is translated in smaller steps, hence finer details can be represented. Conversely, as j decreases, $\varphi_{j,k}$ becomes wider and is translated in larger steps, hence only coarser information can be represented and the span is smaller.

Multiresolution Analysis

The basic requirement for multi-resolution analysis (MRA) [20] is a nesting of the spanned spaces such that

$$\mathcal{V}_j \subset \mathcal{V}_{j+1}, \quad \forall j \in \mathbb{Z}, \quad (2.53)$$

with

$$\mathcal{V}_{-\infty} = \{0\}, \quad \mathcal{V}_{\infty} = L^2(\mathbb{R}). \quad (2.54)$$

Since $\varphi \in \mathcal{V}_0$, it is also in $\mathcal{V}_1$, the space spanned by $\{\varphi_{1,k}\}_{k \in \mathbb{Z}}$. This means that φ can be expanded over $\mathcal{V}_1$ as

$$\varphi(t) = \sqrt{2} \sum_{n \in \mathbb{Z}} h(n) \varphi(2t - n), \quad (2.55)$$

where $h(n)$ is the scaling coefficients (or scaling filter) and $\sqrt{2}$ maintains the norm of the scaling function with the scale of two. This *recursive refinement function* is fundamental to the wavelet theory. In fact, $h(n)$ is the LP filter used in the filter bank implementation of wavelet transform.

The Wavelet Functions

Wavelets are a set of functions $\{\psi_{j,k}\}_{j,k \in \mathbb{Z}}$ that span the differences between the spaces spanned by the various scales of the scaling function. It is often advantageous to require the scaling functions and wavelets be orthogonal; for example, Parseval's theorem holds for orthonormal wavelets. Let the orthogonal complement of $\mathcal{V}_j$ in $\mathcal{V}_{j+1}$, denoted as $\mathcal{W}_j$, be spanned by $\{\psi_{j,k}\}_{k \in \mathbb{Z}}$ such that

$$\langle \varphi_{j,l}, \psi_{j,k} \rangle = 0, \quad (2.56)$$

for all appropriate $j, k, l \in \mathbb{Z}$.

It follows that

$$\begin{aligned} \mathcal{V}_1 &= \mathcal{V}_0 \oplus \mathcal{W}_0, \\ \mathcal{V}_2 &= \mathcal{V}_0 \oplus \mathcal{W}_0 \oplus \mathcal{W}_1, \\ &\dots \\ L^2 &= \mathcal{V}_0 \oplus \mathcal{W}_0 \oplus \mathcal{W}_1 \oplus \dots \end{aligned} \quad (2.57)$$

where $\mathcal{V}_0$ is the initial space spanned by the scaling functions $\{\varphi_{0,k}\}_{k \in \mathbb{Z}}$, and $\oplus$ denotes union of spaces. If the scale of the initial space is chosen to be $j = -\infty$, we have

$$L^2(\mathbb{R}) = \dots \oplus \mathcal{W}_{-2} \oplus \mathcal{W}_{-1} \oplus \mathcal{W}_0 \oplus \mathcal{W}_1 \oplus \mathcal{W}_2 \oplus \dots \quad (2.58)$$

Since the wavelet $\psi_{0,0}$ is also in the space $\mathcal{V}_1$ spanned by the next narrower scaling functions $\{\varphi_{1,k}\}_{k \in \mathbb{Z}}$, it can be expanded as

$$\psi_{0,0}(t) \equiv \psi(t) = \sqrt{2} \sum_{n \in \mathbb{Z}} g(n) \varphi(2t - n), \quad (2.59)$$

where $g(n)$ is the wavelet coefficients and $\sqrt{2}$ maintains the norm of the wavelet function of the scale of two. In fact, $g(n)$ is the HP filter used in the filter bank implementation of wavelet transform.

The function ψ generated by (2.59) is called the *generating wavelet* or *mother wavelet*, from which expansion functions over space $\mathcal{W}_j$ are complementary to $\mathcal{V}_j$ in $\mathcal{V}_{j+1}$ and can be derived:

$$\psi_{j,k}(t) = 2^{j/2} \psi(2^j t - k). \quad (2.60)$$

With a set of functions $\{\varphi_k\}_{k \in \mathbb{Z}}$ and $\{\psi_{j,k}\}_{j,k \in \mathbb{Z}}$ that span all of $L^2(\mathbb{R})$, according to (2.57), any function $f \in L^2(\mathbb{R})$ can be written as a series expansion in terms of the scaling function and wavelets:

$$f(t) = \sum_{k=-\infty}^{\infty} a_k \varphi_k(t) + \sum_{j=0}^{\infty} \sum_{k=-\infty}^{\infty} d_{j,k} \psi_{j,k}(t). \quad (2.61)$$

Biorthogonal Wavelets

The biorthogonal wavelets require *two* scaling functions and *two* wavelet functions [21][27]. The scaling function φ and the dual scaling function $\tilde{\varphi}$ are defined as

$$\varphi(t) = \sqrt{2} \sum_{n \in \mathbb{Z}} h(n) \varphi(2t - n), \quad (2.62)$$

$$\tilde{\varphi}(t) = \sqrt{2} \sum_{n \in \mathbb{Z}} \tilde{h}(n) \tilde{\varphi}(2t - n), \quad (2.63)$$

where $h(n)$ is the scaling coefficients, and $\tilde{h}(n)$ is the dual scaling coefficients. The wavelet and dual wavelet are defined as

$$\psi(t) = \sqrt{2} \sum_{n \in \mathbb{Z}} g(n) \varphi(2t - n), \quad (2.64)$$

$$\tilde{\psi}(t) = \sqrt{2} \sum_{n \in \mathbb{Z}} \tilde{g}(n) \tilde{\varphi}(2t - n), \quad (2.65)$$

where $g(n)$ is the wavelet coefficients and $\tilde{g}(n)$ the dual wavelet coefficients.

We can expand functions using the scaling and wavelet functions and reconstruct them using their duals. Similar to the orthogonal case, at scale j , four spaces can be defined as follows

$$\mathcal{V}_j = \overline{\text{span}\{\varphi_{j,k}\}_{k \in \mathbb{Z}}}, \quad (2.66)$$

$$\tilde{\mathcal{V}}_j = \overline{\text{span}\{\tilde{\varphi}_{j,k}\}_{k \in \mathbb{Z}}}, \quad (2.67)$$

$$\mathcal{W}_j = \overline{\text{span}\{\psi_{j,k}\}_{k \in \mathbb{Z}}}, \quad (2.68)$$

$$\tilde{\mathcal{W}}_j = \overline{\text{span}\{\tilde{\psi}_{j,k}\}_{k \in \mathbb{Z}}}, \quad (2.69)$$

for which, we have

$$\mathcal{V}_j \perp \tilde{\mathcal{W}}_j, \quad \tilde{\mathcal{V}}_j \perp \mathcal{W}_j. \quad (2.70)$$

The multi-resolution nesting relationship in (2.53) is revised accordingly as

$$\mathcal{V}_j \subset \mathcal{V}_{j+1}, \quad \tilde{\mathcal{V}}_j \subset \tilde{\mathcal{V}}_{j+1}, \quad \forall j \in \mathbb{Z}. \quad (2.71)$$

(a) Analysis filter bank for forward wavelet transform

(b) Synthesis filter bank for inverse wavelet transform

Figure 2.3: Filter bank implementation of forward and inverse orthogonal wavelet transforms. The scaling coefficients and wavelet coefficients are used by the synthesis FB for inverse WT, and their time-reversed versions are used by the analysis FB for forward WT.

2.3.1 Computation of Wavelet Transform Using Filter Banks

Mallat showed [20] that both the scaling and wavelet coefficients at scale j can be obtained by passing the scaling coefficients at scale $j + 1$ through an analysis filter bank with one filter derived from the recursive equation governing the scaling function and the other filter derived from the equation governing the wavelet function and scaling function. The reconstruction of the original coefficients at scale $j + 1$ is done by passing the scaling and wavelet coefficients at scale j through a synthesis filter bank which is related to the analysis filter bank as described in Section 2.2.

Specifically, for orthogonal wavelets, we can implement the forward and inverse transforms with the two-band filter bank depicted in Fig. 2.1, with $h_0(n) = h(-n)$, $h_1(n) = g(-n)$, $g_0(n) = h(n)$, and $g_1(n) = g(n)$. This is illustrated in Fig. 2.3, where a_j is the scaling coefficients and d_j is the wavelet coefficients, both at scale j . For the biorthogonal case, we choose the filters such that $h_0(n) = h(-n)$, $h_1(n) = g(-n)$, $g_0(n) = \tilde{h}(n)$, and $g_1(n) = \tilde{g}(n)$.

Iterating the filter bank on the scaling coefficients gives the wavelet coefficients at multiple scales and the scaling coefficients at the coarsest scale, as in illustrated Fig. 2.4(a). The corresponding reconstruction is shown in Fig. 2.4(b). This iteration results in a logarithmic division of the spectrum as shown in Fig. 2.5. It is interesting to note that the ratio of the bandwidth to the center frequency of the band is constant. The collection of the wavelet coefficients and the scaling coefficients gives a multi-resolution representation of the original signal. In the example of Fig. 2.4, the coefficients in $\mathcal{V}_0$ give the coarsest representation of the original signal, those in $\mathcal{V}_0$ and those in $\mathcal{W}_0$ together give one level higher resolution, and so forth.

(a) Analysis tree

(b) Synthesis tree

Figure 2.4: Two-band filter bank iteration tree. Only three stages are shown for clarity.

Figure 2.5: Frequency bands for the analysis tree

Figure 2.6: 2D wavelet transform with 1D wavelet filters. The LP filter and HP filter are labelled L and H , respectively. A is the approximation signal, D_V contains the details in the vertical direction, D_H the details in the horizontal direction, and D_D the details in both or diagonal directions.

2.3.2 Multi-Dimensional Wavelet Transform

Wavelet expansion of functions in $L^2(\mathbb{R}^N)$ is obtained by a separable extension of the one-dimensional decomposition algorithm described previously. For instance, the two-dimensional wavelet transform can be computed first along the columns then along the rows, or first along the rows then along the columns, as illustrated in Fig. 2.6. The original 2D signal is decomposed into four sub-signals, with one sub-signal being the approximation and the other three being details in different directions. In the FB terminology, the approximation sub-signal A is often referred as the ‘LL’ (low-low) band as it is the result of LP filtering in both horizontal and vertical directions, and the detail sub-signals D_V , D_H , D_D as ‘HL’, ‘LH’ and ‘HH’ bands, respectively. To illustrate the directionality of 2D wavelet transform, Fig. 2.7 shows the 2D wavelet transform of an artificial image, where the vertical edges are visible in the HL (D_V) band, horizontal edges visible in the LH (D_H) band, and diagonal edges visible in the HH (D_D) band.

2.4 Lifting Factorization of Wavelet Transform

We now describe the lifting factorization of wavelet transform that is often used in signal processing applications. The key idea of lifting is to realize complicated transforms with a cascade of simple and reversible stages. Here we present an easy (and somewhat rigid) interpretation of the lifting factorization; strictly correct mathematical details and interpretation can be found in [28][29][30]. The general lifting structure is depicted in Fig. 2.8. The first stage that consists of a unit delay at the lower branch and downsampling at both upper and lower branches, is nothing more than splitting the input signal into two phases with even phase at the upper branch and odd phase at the lower branch. It is also referred to as *lazy* wavelet transform. At each subsequent

Figure 2.7: Illustration of directionality of 2D wavelets. The image in (a) is processed with 9/7 wavelets. The four sub-images in (b) are organized in the order of LL, HL, LH, HH, from left to right, top to bottom.

stage, there are two operations, *lifting* and *dual lifting*. First, the odd phase is *predicted* (*lifted*) from the even phase, and the prediction error becomes the new odd phase signal. Second, the even phase is *updated* (*dual lifted*) with the new odd phase signal (prediction error). These prediction and update operations are denoted by P and U in Fig. 2.8. Finally, the last even phase signal (low-pass band) and the last odd phase signal (high-pass band) are normalized by constants K_L and K_H , respectively, to achieve the desired gain. The inverse transform is realized through inverse operations by first inverting the gains, passing each prediction/update operation with *opposite sign* in exactly the reverse order, and finally merging the two phases with proper relative delay. Since every single step in lifting is reversible, regardless whether or not the prediction operators and the update operators are linear, the whole system is invertible and hence PR is guaranteed.

Figure 2.8: The lifting wavelet transform

Chapter 3

Motion-compensated Wavelet Coding (MCWC)

Since Ohm's pioneering paper in 1994 [14, 31], *motion-compensated wavelet coding* (MCWC) has attracted great attention and significant progress has been made [15, 16, 17, 32, 33, 34, 35, 36, 19, 18]. The emerging MCWC technology, which is in essence a 3-D subband coding approach, represents a major shift from the conventional hybrid coding paradigm. In this chapter, we try to establish a general understanding on MCWC. We will first give an overview of MCWC in Section 3.1, followed by a detailed description of its core functional component, *motion-compensated temporal filtering* (MCTF), in Section 3.2. Section 3.3 provides a brief description of the three major MCWC schemes, namely $t+2D$, $2D+t$, and $2D+t+2D$. As one main motivation of MCWC is the inherent scalable capability, we will analyze and argue the pros and cons of each MCWC scheme in this respect.

3.1 Overview of Motion-compensated Wavelet Coding

Wavelets have been successfully applied in embedded image coding, providing SNR (quality) and spatial (resolution) scalability, e.g., EZW [8], SPIHT [9], EZBC [10] and JPEG2000 [11, 12, 13]. This is due to the simultaneous *spatial-frequency* localization of wavelet transform. Once this key property is efficiently exploited, high coding performance can be achieved. In addition to high coding performance, wavelet image coding techniques usually provide spatial scalability due to the multi-resolution nature of wavelet transform and/or SNR scalability due to successive quantizations employed (e.g., via bitplane coding). Wavelet image coding falls into the general category of 2-D subband coding (2D-SBC). Extending 2-D wavelet coding method for 3-D video signals, we can come up with 3D subband coding (3D-SBC), in which each generated subband corresponds to a particular frame rate and spatial scale and is represented by an embedded bitstream. One distinctive feature of 3D-SBC is its inherent ability of offering temporal, spatial and SNR scalability simultaneously.

Direct attempt to applying three-dimensional wavelets to video coding has been made, achieving results comparable to that of MPEG-2 [37]. However, it is well known that the temporal

Figure 3.1: The general 2D+t+2D framework

correlation can not be maximally exploited without motion compensation, because moving objects are most correlated along their motion paths. It is thus necessary to combine motion compensation and wavelet transform for temporal analysis, which is termed *motion-compensated temporal filtering* (MCTF) in the literature. 3-D wavelet video coding employing MCTF is thus referred to as *motion-compensated wavelet coding* (MCWC).

A natural question arising from 3D wavelet transform in video coding is in which order to perform the transform. In the conventional prediction-based coders like MPEG codecs, motion-compensated residue is first generated and then spatially transformed by some 2D transformation (e.g., 2D DCT), which may be viewed as the *temporal-spatial* order. In MCWC, the video signal can be transformed in either the temporal-spatial order or the *spatial-temporal* order. In the literature, the former case is widely referred to as a t+2D scheme, and the latter as a 2D+t scheme. Putting both t+2D and 2D+t schemes in a generic framework, we obtain the most general 2D+t+2D scheme [38, 39]: spatial subbands are first generated by performing 2-D discrete wavelet transform (2D-DWT) (*pre-spatial* wavelet decomposition), then MCTF is performed on these spatial subbands, and at last 2D-DWT further decomposes the obtained spatiotemporal subbands for removal of remaining spatial redundancy (*post-spatial* wavelet decomposition). Fig. 3.1 depicts such a general framework that can function as a t+2D scheme when the pre-spatial decomposition is absent, or that can also function as a 2D+t scheme when the post-spatial decomposition is removed.

Regardless of the order of temporal and spatial decompositions, MCTF is carried out along the temporal direction of the input frames in a pyramidal fashion. For illustration, consider a three-level dyadic analysis on a group of eight consecutive input frames, denoted as $F_i, 0 \leq i \leq 7$, as shown in Fig. 3.2. At the top level, the original sequence is filtered and (maximally) downsampled into one low-pass band and one high-pass band with four frames each, denoted by $L_i^1, 0 \leq i \leq 3$ for the low-pass frames and by $H_i^1, 0 \leq i \leq 3$ for the high-pass frames, where the superscript 1 indicates the first level of temporal decomposition and subscript i indicates the frame number (time index) in the decomposed domain. Note that the low-pass frames resemble a time-averaged version of the input video at half frame rate, while the high-pass frames resemble the variation along the motion paths. MCTF is then recursively performed on the temporal low-pass band, generating high-pass bands H_i^2 ($i = 0, 1$), H_i^3 ($i = 0$) and low-pass band L_i^3 ($i = 0$). After multiple levels of MCTF,

Figure 3.2: The spatiotemporal wavelet decomposition tree. The spatial wavelet decomposition is performed on the leaf frames, indicated by the 2D dyadic partition grid. The intermediate temporal low-pass frames are not part of the final representation, and hence no spatial decomposition is performed.

a tree of temporal frames are generated, where the leaf frames constitute the final representation and the node frames, i.e., the temporal low-pass frames above the lowest temporal level, have been decomposed in the process of MCTF and hence are not part of the final representation. Due to the dyadic nature of wavelet analysis conducted in MCTF, the number of high-pass frames at temporal level j is $N/2^j$, $1 \leq j \leq M$, where N is the number of original input frames, M the total levels of MCTF and it is assumed for clarity that $N = a2^M$ for a positive integer a . Note that the lowest temporal low-pass band L^M is part of the final representation, which has $N/2^M = a$ frames. The original input sequence is considered as temporal band L^0 . Temporal low-pass band L^j , $0 \leq j < M$, can be reconstructed based on its descendent bands $L^M, H^M, H^{M-1}, \dots, H^{j+1}$, which corresponds to the input video at the frame rate of $1/2^j$. It is also interesting to note that even when temporal scalability is not required, it may also be advantageous to perform multi-level MCTF to exploit temporal correlations for high compression performance, especially for those video sequences with low-motion contents.

Spatial wavelet decomposition is performed on the leaf frames of the temporal decomposition tree to exploit the spatial correlations in each frame. The number of levels of spatial decomposition depends on the spatial correlations existing in these frames as well as the number of spatial resolutions to be supported in the final bitstream. Suppose S spatial resolutions are to be supported. Then, at least $S - 1$ levels of spatial decomposition have to be performed for all these frames. Furthermore, since the temporal low-pass frames often exhibit high spatial correlations, they may be decomposed by more levels for better coding efficiency than the temporal high-pass frames that in general are not highly spatially correlated.

The final spatiotemporal subbands after 3D wavelet decomposition are subject to entropy coding. Since these subband images are in the various sizes (due to the maximally downsampled wavelet decomposition), embedded image coding techniques (e.g., [9, 10, 13]) are often employed to entropy code the subbands either jointly or independently. In either case, an embedded bitstream is generated to support SNR scalability. For a brief introduction to embedded wavelet image coding, readers are referred to Appendix A.

One major difference between MCTF and *motion-compensated prediction* (MCP) exploited in conventional hybrid coders [1, 2, 3] is that MCTF assumes an *open-loop* structure, as there is no feedback of previously coded data into the coding process of the current frame/subband. As long as the temporal filter used for MCTF is of *finite* length (i.e., FIR filter), the quantization distortion in one frame will be limited within a certain number of frames. This eliminates the drift problem encountered in the conventional hybrid coders. When orthonormal or nearly orthonormal wavelets are used for decomposition, the distortion due to quantization of spatiotemporal subband coefficients can be directly estimated in the subband domain, without considering the range of affected frames.

3.2 Motion-compensated Temporal Filtering (MCTF)

It is well-known that highest compression of video signals results only when both spatial and temporal redundancies are removed. Temporal redundancy mostly exists along the moving paths of objects, particularly incurred in the case of translational motion. Thus, to maximally remove the

temporal redundancy in a video sequence, wavelet analysis in the temporal domain should be applied along the motion paths, i.e., *motion-compensated temporal filtering* (MCTF) [18, 19, 33]. Through this way, the temporal high-pass frames will have much smaller energy (i.e., prediction error) than that without motion compensation (MC). According to the well-established rate-distortion (R-D) theory, to achieve a certain mean squared error distortion D for a Gaussian source with zero mean and variance σ^2 , the bitrate has a lower bound given by [40]:

$$R(D) = \frac{1}{2} \log_2 \frac{\sigma^2}{D} \quad (3.1)$$

Thus, temporal filtering with MC will yield a reduction in total bitrate because temporal high-pass frames will have a reduced variance, as long as the cost of transmitting motion vectors does not exceed the reduction in bits allocated for the high-pass frames. Hereby, we would like to point out that MCTF typically does not mandate any particular motion estimation/compensation scheme, although block-matching motion estimation (BMME) seems to be most widely used due to its good trade-off between efficiency and complexity.

Fig. 3.3 shows some results of temporal filtering both without and with motion compensation. It is clearly seen that without motion compensation the low-pass frame is heavily blurred, while the high-pass frame contains high energy. In contrast, when motion compensation is used in temporal filtering, the low-pass frame retains most information of the two input frames with clear and sharp content, as a result, the high-pass frame has low energy. This example clearly shows that MC is quite effective for temporal filtering to reduce the energy of high-pass frames for better coding performance. Therefore, the low-pass frames without MC are a poor approximation of the video sequence at a lower frame rate. It is worse than directly subsampling the input video.

3.2.1 How MCTF Operates

To temporally filter a video sequence along the motion paths, pixels from different frames need to be aligned before convolving with the temporal filter. If the filter length is K , pixels from other $K - 1$ frames need to be aligned with the pixel from the current frame, in order to produce one filtered pixel at the current frame time. Obviously, this alignment is governed by the motion vectors found through some motion estimation scheme, e.g., BMME.

To illustrate how MCTF operates, let us consider the case of two-tap temporal filter for the sake of simplicity. Denote A as the input frame at time $2t$ and B as the input frame at time $2t + 1$. Furthermore, let the temporal low-pass frame be L and high-pass frame be H . We choose frame A as the reference from which we predict B by exploiting BMME, so that we obtain a forward motion vector field (MVF). Fig. 3.4(a) illustrates the MVF from A to B . When there is a one-to-one correspondence between pixels in B and those in A , we classify them as *connected* [14, 4]. Pixels in A that are not connected to any pixel in B are classified as *unconnected*, or *covered* to be more specific, as they only exist at an earlier time. The pixels labelled 1 and 2 in Fig. 3.4(a) fall into the *covered* category. On the other hand, there may be multiple pixels in B which are connected to the same pixel in A ; in this case we also classify them as *unconnected*, or *uncovered* as they only appear at a later time, *except* one of them. This exceptional pixel is actually classified as *connected*, which

(a) Low-pass frame without MC

(b) Low-pass frame with MC

(c) High-pass frame without MC

(d) High-pass frame with MC

Figure 3.3: Temporally filtered frames with and without motion compensation (generated by the MC-EZBC [17] coder whose source code was obtained from the MPEG CVS server on 20 Dec. 2005).

Figure 3.4: Motion connection classification and corresponding temporal filtering. ‘ a ’ represents a pixel value from frame A , and ‘ b ’ a pixel value from frame B .

may be determined by, for example, a raster scan method. In Fig. 3.4(a), the pixels labelled 3, 4, 5 and 6 are multiple-connected to pixels in the reference frame, and 3 and 4 are eventually classified as *connected* (according to some criteria) while 5 and 6 classified as *uncovered*. Note that when BMME is used, the number of *covered* pixels and that of *uncovered* pixels are equal. However, they are not equal if the motion estimation algorithm links groups of pixels that are of different sizes, e.g., deformable mesh motion model [41, 42, 43]. In general, when there exists motion other than translational type such as occlusion, scaling, rotation and so on, the motion field is inhomogeneous and isolated (uncovered or covered) areas may be resulted.

When applying MCTF to A and B , we align the temporal high-pass frame H with frame B and temporal low-pass frame L with frame A ; that is, positions in B are mapped into identical positions in H , and positions in A into the identical positions in L . We shall employ the orthonormal Haar wavelet whose low-pass filter is given by $\{\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\}$ and high-pass filter by $\{\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\}$. Let (d_m^B, d_n^B) denote the motion vector found for pixel $B[m, n]$, where m denotes the horizontal position and n the vertical position. When integer-accurate motion vectors are used, the temporal high-pass frame H can be generated, regardless of the connection type (see Fig. 3.4(b)), as

$$H[m, n] = \frac{1}{\sqrt{2}}B[m, n] - \frac{1}{\sqrt{2}}A[m + d_m^B, n + d_n^B] \quad (3.2)$$

Remark: The applicability of (3.2) to the *connected* pixels in B is obvious. For those *unconnected/uncovered* pixels, *displaced frame differences* (DFD) are usually substituted into H , scaled by some factor to maintain the dynamic range. Note that these *uncovered* pixels have also been found with matching pixels in A ; thus, it is convenient to treat them in the same way as the *con-*

connected ones. Through this way, the high-pass frame H is in fact the MCP residue generated in conventional hybrid coders.

The *connected* pixels in A produce L such that

$$L[m + d_m^B, n + d_n^B] = \frac{1}{\sqrt{2}}A[m + d_m^B, n + d_n^B] + \frac{1}{\sqrt{2}}B[m, n] \quad (3.3)$$

Note that (3.3) also applies to the multiple-connected pixels in A , by selecting the *connected* pixel in B . The gray circles in Fig. 3.4(b) represent *uncovered* pixels in B that are not selected for generating L ; these pixels are only used for generating H , which is indicated by the one-headed arrow.

The *unconnected/covered* pixels in A are directly substituted into L with a scaling factor of $\sqrt{2}$ (see Fig. 3.4(b)), i.e.,

$$L[m, n] = \sqrt{2}A[m, n] \quad (3.4)$$

Remark: It can be observed that (3.2), (3.3), and (3.4) form a system of two equations with two unknowns A and B . Therefore, it is possible to solve for A and B from L , H and MVs at the decoder. That is, perfect reconstruction (PR) is achievable with *integer*-accurate MVs.

On the other hand, if the motion vectors are *subpixel*-accurate, the above definition of connection should be revised slightly. We state that pixel $A[m + \llbracket d_m^B \rrbracket, n + \llbracket d_n^B \rrbracket]$ is *connected* to $B[m, n]$, where $\llbracket \cdot \rrbracket$ denotes the rounding operator that rounds a number to its nearest integer. Let $\hat{A}$ denote an interpolated version of A , and similarly $\hat{B}$ an interpolated version of B . The high-pass frame H is then generated by

$$H[m, n] = \frac{1}{\sqrt{2}}B[m, n] - \frac{1}{\sqrt{2}}\hat{A}[m + d_m^B, n + d_n^B] \quad (3.5)$$

The pixels of the low-pass L frame which correspond to the *connected* positions in A are given by

$$L[m + \llbracket d_m^B \rrbracket, n + \llbracket d_n^B \rrbracket] = \frac{1}{\sqrt{2}}A[m + \llbracket d_m^B \rrbracket, n + \llbracket d_n^B \rrbracket] + \frac{1}{\sqrt{2}}\hat{B}[m + \llbracket d_m^B \rrbracket - d_m^B, n + \llbracket d_n^B \rrbracket - d_n^B] \quad (3.6)$$

And *unconnected* pixels of L are treated as in (3.4).

Remark: Again, it can be observed that (3.5), (3.6), and (3.4) form a system of two equations with four unknowns A , $\hat{A}$, B and $\hat{B}$. It is thus impossible to perfectly reconstruct A and B from L , H and MVs at the decoder. Although $\hat{A}$ and $\hat{B}$ may be obtained by *linear* interpolation at subsampling positions of A and B , respectively, the actual interpolation position (phase) depends on the (reverse) motion vector. Hence the interpolation can be viewed as a *spatially varying* filtering, which is not reversible.

The non-PR property for subpixel-accurate MVs is a serious drawback of the implementation described above. Fortunately, it is possible to achieve PR with subpixel-accurate MVs by the *lifting* implementation of wavelet transform. Recall that, as described in Section 2.4, the lifting implementation is a series of paired prediction/update operations in cascade, and even if the prediction/update operators are non-linear, lifting wavelet transform is still reversible. The high-pass and low-pass filtering as well as the associated (inverse) MC in MCTF can be implemented in such a lifting framework, which eventually makes MCTF reversible. In the following two subsections,

we will look in detail at the lifting implementation of two most common wavelets used in MCTF, namely Haar wavelet and biorthogonal 5/3 wavelet.

3.2.2 MCTF with Lifting Haar Transform

Let us first establish some notations for ease of description. Unless explicitly stated, we will assume that the temporal high-pass frames are aligned with odd-indexed frames, and low-pass frames with even-indexed frames. Due to this alignment, motion vectors are estimated from even-indexed frames (as the reference frames) to odd-indexed frames (as the current frames); for example, in the case of two-tap Haar transform, motion vectors are estimated from frame $2k$ to frame $2k + 1$. Let $X_k[\mathbf{n}]$ denote a pixel at position $\mathbf{n}$ in frame k , where $\mathbf{n}$ is a 2-D vector representing coordinates. Let $\hat{X}_k$ be an interpolated version of X_k . Denote $d_{\mathbf{n}}^{2k \rightarrow 2k+1}$ as the motion vector for pixel $\mathbf{n}$ of X_{2k+1} that is estimated from X_{2k} . Consequently, $d_{\mathbf{n}}^{2k+1 \rightarrow 2k}$ will denote the associated inverse motion vector for pixel $\mathbf{n}$ of X_{2k} , which points to some location in X_{2k+1} . Note that the inverse motions vectors need not be found by motion estimation; they can be derived through connections as discussed in the previous section and illustrated in Fig. 3.4. As we assume the motion vectors are subpixel-accurate, they may point to subpixel positions in the reference frame. Thus, the definitions of *connected* and *disconnected* pixels should be revised slightly. The pixel in the reference frame (X_{2k}) that is *nearest* to the subpixel position pointed by a motion vector is considered to be *connected*, while others considered to be *unconnected*. With connection established in this way, the former definitions of *covered* and *uncovered* pixels can easily be adapted to subpixel-accurate motion vectors.

MCTF using the lifting Haar transform is illustrated in Fig. 3.5. The lifting Haar analysis for high-pass frame can be expressed as:

$$H_k[\mathbf{n}] = \frac{1}{\sqrt{2}} X_{2k+1}[\mathbf{n}] - \frac{1}{\sqrt{2}} \hat{X}_{2k}[\mathbf{n} + d_{\mathbf{n}}^{2k \rightarrow 2k+1}]. \quad (3.7)$$

For *connected* pixels in X_{2k} , the lifting Haar analysis for low-pass frame is expressed as

$$L_k[\mathbf{n}] = \sqrt{2} X_{2k}[\mathbf{n}] + \hat{H}_k[\mathbf{n} + d_{\mathbf{n}}^{2k+1 \rightarrow 2k}], \quad (3.8)$$

where, it should be noted that, the inverse motion vector is used. For *unconnected* pixels in X_{2k} , they are directly substituted into L_k :

$$L_k[\mathbf{n}] = \sqrt{2} X_{2k}[\mathbf{n}]. \quad (3.9)$$

The lifting synthesis is straightforwardly realized by reversing the operations of analysis. Frame X_{2k} is reconstructed first, by inverting the update operation. For frame X_{2k} , the *connected* pixels are reconstructed by

$$X_{2k}[\mathbf{n}] = \frac{1}{\sqrt{2}} L_k[\mathbf{n}] - \frac{1}{\sqrt{2}} \hat{H}_k[\mathbf{n} + d_{\mathbf{n}}^{2k+1 \rightarrow 2k}], \quad (3.10)$$

and the *unconnected* pixels reconstructed by

$$X_{2k}[\mathbf{n}] = \frac{1}{\sqrt{2}} L_k[\mathbf{n}]. \quad (3.11)$$

Figure 3.5: MCTF with lifting Haar transform. $W_{i \rightarrow j}$ represents motion mapping from frame i to frame j .

Then, with the newly reconstructed X_{2k} , frame X_{2k+1} is reconstructed by

$$X_{2k+1}[\mathbf{n}] = \sqrt{2}H_k[\mathbf{n}] + \hat{X}_{2k}^{2k \rightarrow 2k+1}[\mathbf{n}]. \quad (3.12)$$

Remark: The prediction operator in this case is in fact a spatially varying filter (interpolator) according to MVs, and so is the update operator (but according to inverse MVs). As can be easily verified, when integer-accurate MVs are used, the lifting implementation is equivalent to that described in the previous section. However, when subpixel-accurate MVs are used, in the lifting implementation the low-pass frame L is generated using interpolated H instead of interpolated B/X_{2k+1} (current frame). Similar argument applies for generating the high-pass frame H . In this way, the decoder sees the same information as the encoder did, avoiding introducing additional unknowns ($\hat{A}$ and $\hat{B}$) into the system of equations. Hence PR is guaranteed.

3.2.3 MCTF with Lifting 5/3 Transform

Now let us take a look at MCTF with the more complex biorthogonal 5/3 wavelet, whose low-pass filter is given by the transfer function $H_l(z) = -\frac{1}{8}z^2 + \frac{1}{4}z + \frac{3}{4} + \frac{1}{4}z^{-1} - \frac{1}{8}z^{-2}$ and high-pass filter by $H_h(z) = -\frac{1}{2}z + 1 - \frac{1}{2}z^{-1}$. Their lifting structures are shown in Fig. 3.6, respectively.

With the same notations as adopted previously, the analysis of the high-pass frame can be expressed as:

$$H_k[\mathbf{n}] = X_{2k+1}[\mathbf{n}] - \frac{1}{2}\hat{X}_{2k}[\mathbf{n} + d_{\mathbf{n}}^{2k \rightarrow 2k+1}] - \frac{1}{2}\hat{X}_{2k+2}[\mathbf{n} + d_{\mathbf{n}}^{2k+2 \rightarrow 2k+1}]. \quad (3.13)$$

For the low-pass frame, there are four cases:

(a) Analysis

(b) Synthesis

Figure 3.6: MCTF with lifting biorthogonal 5/3 transform. $W_{i \rightarrow j}$ represents motion mapping from frame i to frame j .

- *Bidirectionally connected* pixels, for which X_{2k} is connected to both X_{2k} and X_{2k+1}

$$L_k[\mathbf{n}] = X_{2k}[\mathbf{n}] + \frac{1}{4}\hat{H}_{k-1}[\mathbf{n} + d_{\mathbf{n}}^{2k-1 \rightarrow 2k}] + \frac{1}{4}\hat{H}_k[\mathbf{n} + d_{\mathbf{n}}^{2k+1 \rightarrow 2k}]. \quad (3.14)$$

- *Forward connected* pixels, for which X_{2k} is connected to X_{2k+1} only

$$L_k[\mathbf{n}] = X_{2k}[\mathbf{n}] + \frac{1}{4}\hat{H}_k[\mathbf{n} + d_{\mathbf{n}}^{2k+1 \rightarrow 2k}]. \quad (3.15)$$

- *Backward connected* pixels, for which X_{2k} is connected to X_{2k-1} only

$$L_k[\mathbf{n}] = X_{2k}[\mathbf{n}] + \frac{1}{4}\hat{H}_{k-1}[\mathbf{n} + d_{\mathbf{n}}^{2k-1 \rightarrow 2k}]. \quad (3.16)$$

- *Unconnected* pixels, for which X_{2k} is neither connected to X_{2k-1} nor to X_{2k+1} .

$$L_k[\mathbf{n}] = X_{2k}[\mathbf{n}]. \quad (3.17)$$

Lifting 5/3 wavelet synthesis for *bidirectionally connected*, *forward connected*, *backward connected* and *unconnected* pixels of X_{2k} , respectively, is given by

$$X_{2k}[\mathbf{n}] = L_k[\mathbf{n}] - \frac{1}{4}\hat{H}_{k-1}[\mathbf{n} + d_{\mathbf{n}}^{2k-1 \rightarrow 2k}] - \frac{1}{4}\hat{H}_k[\mathbf{n} + d_{\mathbf{n}}^{2k+1 \rightarrow 2k}], \quad (3.18)$$

$$X_{2k}[\mathbf{n}] = L_k[\mathbf{n}] - \frac{1}{4}\hat{H}_k[\mathbf{n} + d_{\mathbf{n}}^{2k+1 \rightarrow 2k}], \quad (3.19)$$

$$X_{2k}[\mathbf{n}] = L_k[\mathbf{n}] - \frac{1}{4}\hat{H}_{k-1}[\mathbf{n} + d_{\mathbf{n}}^{2k-1 \rightarrow 2k}], \quad (3.20)$$

$$X_{2k}[\mathbf{n}] = L_k[\mathbf{n}] \quad (3.21)$$

and synthesis for X_{2k+1} is carried out after frames X_{2k} and X_{2k+2} are reconstructed, as

$$X_{2k+1}[\mathbf{n}] = H_k[\mathbf{n}] + \frac{1}{2}\hat{X}_{2k}[\mathbf{n} + d_{\mathbf{n}}^{2k \rightarrow 2k+1}] + \frac{1}{2}\hat{X}_{2k+2}[\mathbf{n} + d_{\mathbf{n}}^{2k+2 \rightarrow 2k+1}]. \quad (3.22)$$

Remark: Again, the decoder sees the same information as the encoder did, hence the PR property is guaranteed by lifting 5/3 wavelet. It can be seen that the high-pass frames are essentially the bidirectional motion-compensated prediction residues. It has been shown that bidirectional MC reduces residual energy and leads to higher coding performance [33]. However, compared with MCTF with Haar wavelet where only one reference frame is used for generating each high-pass frame, with 5/3 wavelet the amount of MVs is effectively doubled due to two reference frames exploited to generate each high-pass frame.

3.2.4 Discussion

The temporal low-pass frames can be considered as the weighted averages of several frames, highlighting the information that exists in these frames. MCTF thus provides a powerful means to exploit the temporal correlation over a consecutive number of frames. It is clear that temporal filtering without motion compensation produces heavily blurred low-pass frames and high-energy high-pass frames, resulting in decreased coding efficiency [4, 14].

Temporal correlation typically decreases with time, especially when the sequence contains fast motion. Hence, the level of temporal decomposition should be kept reasonable, otherwise there is a good chance that heavily blurred low-pass frames will result because motion becomes difficult to model precisely as temporal level increases. Another factor for considering the number of temporal decomposition levels is the delay. More levels of temporal decomposition will lead to longer delay and larger memory.

One variation of MCTF is that the update step in lifting may be skipped, resulting in a *truncated* wavelet [44]. For instance, when the update step of 5/3 wavelet is skipped, we equivalently keep the even-indexed frames as low-pass frames, in which case the low-pass filter has an impulse response $h_l(n) = \delta(n)$, or transfer function $H_l(z) = 1$. That is, when truncated, 5/3 wavelet becomes 1/3 wavelet. This truncation can lead to great reduction in computational complexity [45]. Regardless of whether the update step is kept or skipped, however, since the energy of the low-pass frames remains relatively constant, there is little impact on the coding performance of the low-pass frames. Note that this does not mean that the overall coding performance will not be impacted by skipping the update step. In fact, as investigated in [45], skipping the update step will lead to a loss in PSNR performance. The reason is, as argued above, that with the update step the temporal low-pass frames are “averaged” over several frames, therefore preserving more information than the even-indexed frames. This averaging effect could lead to more efficient MC for subsequent levels of MCTF. An alternative explanation is offered by Girod and Han in [46], where the authors, by assuming that both the prediction and update steps are linear operations, show analytically how the update step impacts the distortion in the reconstructed frames.

3.3 Three MCWC Schemes and Their Scalability Issues

In Section 3.1, we mentioned the two distinct schemes of MCWC. We shall study each of these schemes in detail, particularly focusing on their scalability in resolution and frame rate. The SNR scalability is inherently provided by the embedded texture and entropy coding method employed, and it is in general not related to the decomposition structure used. Therefore, the SNR scalability is not in the scope of discussion in this section.

One question related to measuring the distortion of the video sequence reconstructed at a lower spatial resolution and/or a lower frame rate is how to generate the reference video sequence of that particular spatial resolution and frame rate. For temporal scalability, it is trivial to produce the lower frame rate reference by direct downsampling, i.e., selecting the frames at proper time instants. Alternatively, the reference video may be taken as the output of the update step at the proper level of MCTF. For spatial scalability, it should be noted that the LL band generated by the spatial decomposition contains some amount of high frequency aliasing, due to the relatively short spatial filters (such as 9/7 wavelets) employed in the coder. However, it is practically convenient to use the LL band as the attainable reference to quantify the PSNR performance of the coder for the lower resolution. This way, it is also possible to compare the PSNR performance for that resolution with that of a non-scalable coder that will take the LL bands as input. Thus, in this thesis we will use the LL band as the lower spatial resolution reference.

In order to produce the reference that is both temporally and spatially scaled, there exist two possibilities: (1) temporally scaled reference video is first generated, and spatial wavelet transform is performed subsequently; and (2) spatially scaled reference video is first generated, followed by temporal scaling. Note that when the trivial method of direct downsampling is used for temporal scaling, the order of performing temporal scaling and spatial scaling does not matter; the same output will result in both cases. However, if some kind of filtering is employed for temporal scaling, e.g., MCTF, the order matters and the output will be different in the two cases. It seems difficult to justify theoretically which order is superior to the other; for evaluation purposes, it seems reasonable and practical to produce the reference by exactly following the 3D decomposition structure of the encoder. In this thesis, the reference video sequence is taken as the spatiotemporal subbands corresponding to the desired resolution and frame rate when they are first generated during 3-D decomposition.

3.3.1 The t+2D Scheme

In the t+2D scheme, MCTF is first performed on the input frames (at the original resolution), then 2D wavelet transform is applied to each of the temporally filtered frames, generating a set of spatiotemporal subbands [16, 33, 17, 18, 47]. Since the temporal filtering is performed in the spatial domain, it is also referred to as *Spatial Domain* MCTF (SD-MCTF). The encoder structure for the t+2D scheme is shown in Fig. 3.7(a). Comparing to the generic framework depicted in Fig. 3.1, the pre-spatial wavelet decomposition is void. Temporal scalability can be achieved simply by skipping the temporal bands above the target frame rate and the associated motion vectors. An example illustrating the encoding process is shown in Fig. 3.8(a).

The decoder reconstructs a video by first spatially inverse transforming the received 3D subbands into the spatial domain, then applying *Inverse* MCTF (IMCTF). Fig. 3.7(b) depicts the decoder structure. When the target spatial resolution is lower than the original, i.e., some spatial high-pass bands are not transmitted to the decoder, IMCTF is conventionally applied at the target (lower) resolution. Since the MVs are estimated at full resolution, they have to be properly down-scaled when applying IMCTF at a lower resolution. Let S denote the total number of spatial resolutions supported by the encoded bitstream and s denote one spatial resolution, where $0 \leq s < S$, $s = 0$ corresponds to the full resolution, $s = 1$ to the half resolution, and so on. Thus, the MVs need to be scaled by a factor of 2^s for IMCTF performed at resolution s . The conventional decoding process for half spatial resolution is depicted in Fig. 3.8(b). In general, the conventional decoding at spatial resolution s can be viewed as a chain of operations:

$$2D-IDWT(S - 1 - s \text{ levels}) \rightarrow IMCTF(\text{with scaled MVs}),$$

which is equivalent to

$$2D-IDWT(S - 1 \text{ levels}) \rightarrow 2D-DWT(s \text{ levels}) \rightarrow IMCTF (\text{with scaled MVs}),$$

where the forward 2D-DWT is to produce frames at the target spatial resolution. As discussed at the beginning of this section, recall that without temporal scaling the reference for a lower spatial resolution is produced by directly applying 2D wavelet transform on the original frames. Hence,

(a) Encoder structure

(b) Decoder structure

Figure 3.7: The t+2D wavelet coding system

ideally, the decoder should do the same, by first reconstructing the full resolution frames and then applying 2D wavelet transform, which can be expressed as

$$2D-IDWT(S - 1 \text{ levels}) \rightarrow IMCTF \text{ (with original MVs)} \rightarrow 2D-DWT(s \text{ levels}).$$

Apparently, in the conventional decoding the order of IMCTF and 2D-DWT is exchanged, apart from that different MVs are used. Among other factors this mismatch between encoder and decoder degrades the PSNR performance for spatially scaled reconstruction, especially at high bitrates. For example, Fig. 3.9 shows the PSNR performance (of MC-EZBC [17]) when decoding the CIF *Foreman* sequence for QCIF resolution. Clearly, PSNR almost stops increasing for bitrates above 1500 kbps. Further analysis of its causes and possible solutions will be provided in Chapter 4.

(a) Encoder

(b) Decoder

Figure 3.8: Encoding and decoding steps for lower resolution video in $t+2D$. The dashed blocks represent data that are not *unavailable* to the decoder.

Figure 3.9: The PSNRs resulted from spatially-scaled decoding in t+2D (generated by the MC-EZBC [17] coder whose source code was obtained from the MPEG CVS server on 20 Dec. 2005).

3.3.2 The 2D+t Scheme

In the 2D+t scheme, input frames are first transformed into spatial subbands by 2D-DWT, and MCTF is subsequently applied to the spatial subbands in each scale. It is also referred to as *in-band* MCTF (IB-MCTF) in the literature, for the MCTF is performed in the spatial subbands. The encoder structure is shown in Fig. 3.10(a). Comparing with the generic framework in Fig. 3.1, the post-spatial decomposition is void because each frame has already been decomposed into dyadic subbands iteratively by the pre-spatial decomposition to provide spatial scalability. Both temporal and spatial scalabilities can be achieved by collecting only the required spatiotemporal subbands with associated motion information and discarding the rest. The decoder, as shown in Fig. 3.10(b), reverses the encoding process with the same information (except quantization noise) as the encoder used. An example illustrating the encoding and decoding processes for spatial scalability in the 2D+t scheme is depicted in Fig. 3.11.

One serious problem with motion compensation in the subband domain in the 2D+t scheme is that maximally downsampled wavelet transform is shift-variant; i.e., a translational movement in the spatial domain does not correspond to a shift in the subband domain unless the shift is an even number of samples. This shift variance is less serious a problem for the lowest LL band, because the low frequencies tend to change slower than high frequencies. Therefore, except for the lowest spatial resolution, the PSNR performance is compromised if MC is constrained to the same

Figure 3.10: The 2D+t wavelet coding system

subband. To overcome this drawback, algorithms, such as the low-band shift (LBS) method [48] and phase-shifting filters [49], have been proposed to recover from the critically sampled wavelet data the phase information discarded by maximal downsampling. These methods are generally referred to as a *complete-to-overcomplete wavelet transform* (CODWT) [50], as an overcomplete representation is generated from a complete representation.

To give readers a more concrete idea of the relationships between the various phases of subband signals, we make use of the result in [49] and lay down explicitly the formulas for computing the phases discarded by downsampling. In [49], it is shown that for a two-band one-dimensional (1D) subband decomposition, the odd phase (discarded by downsampling) of the low-pass and high-pass subbands, denoted as S_0 and D_0 respectively, is related to the even phase (retained by downsampling) of the low-pass and high-pass subbands, denoted as S_1 and D_1 respectively, by a *phase-shifting matrix* such that

$$\begin{bmatrix} S_1 \\ D_1 \end{bmatrix} = T \begin{bmatrix} S_0 \\ D_0 \end{bmatrix} \quad (3.23)$$

where T is the phase-shifting matrix that is given by

$$T = \frac{1}{2} \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} \quad (3.24)$$

(a) Encoder

(b) Decoder

Figure 3.11: Encoding and decoding steps for lower resolution video in 2D+t. The dashed blocks represent *unavailable* data.

where

$$\begin{aligned}
T_{11} &= H_0(z^{1/2})H_1(-z^{1/2}) + H_0(-z^{1/2})H_1(z^{1/2}) \\
T_{12} &= -2H_0(z^{1/2})H_0(-z^{1/2}) \\
T_{21} &= 2H_1(z^{1/2})H_1(-z^{1/2}) \\
T_{22} &= -T_{11}
\end{aligned}$$

and H_0 and H_1 are the low-pass and high-pass filters, respectively.

In the two-dimensional case, there are four phases in each subband. Let $LL_{i,j}$, $HL_{i,j}$, $LH_{i,j}$ and $HH_{i,j}$ ($i \in \{0,1\}, j \in \{0,1\}$) denote phase (i,j) of the LL, HL, LH, and HH subbands, respectively. Then we need to obtain three missing phases for each subband from the (0,0) phase signals that are retained after downsampling. Note that the implicit assumption here is that the 2D subband decomposition is performed by two separable 1D subband decompositions, which is often the case in practice. To distinguish the phase shifting operation in the horizontal and vertical directions, we use T_h to denote the operation in the horizontal direction, and use T_v , $\begin{bmatrix} T_{11} \\ T_{12} \end{bmatrix}_v$ and $\begin{bmatrix} T_{21} \\ T_{22} \end{bmatrix}_v$ to denote operations in the vertical direction. Thus, we have

$$\begin{bmatrix} LL_{0,1} \\ LH_{0,1} \end{bmatrix} = T_v \begin{bmatrix} LL_{0,0} \\ LH_{0,0} \end{bmatrix} \quad (3.25)$$

$$\begin{bmatrix} HL_{0,1} \\ HH_{0,1} \end{bmatrix} = T_v \begin{bmatrix} HL_{0,0} \\ HH_{0,0} \end{bmatrix} \quad (3.26)$$

$$\begin{bmatrix} LL_{1,0} \\ HL_{1,0} \end{bmatrix} = T_h \begin{bmatrix} LL_{0,0} \\ HL_{0,0} \end{bmatrix} \quad (3.27)$$

$$\begin{bmatrix} LH_{1,0} \\ HH_{1,0} \end{bmatrix} = T_h \begin{bmatrix} LH_{0,0} \\ HH_{0,0} \end{bmatrix} \quad (3.28)$$

$$\begin{bmatrix} LL_{1,1} \\ HL_{1,1} \end{bmatrix} = T_h \begin{bmatrix} LL_{0,1} \\ HL_{0,1} \end{bmatrix} = T_h \begin{bmatrix} LL_{0,0} & LH_{0,0} \\ HL_{0,0} & HH_{0,0} \end{bmatrix} \begin{bmatrix} T_{11} \\ T_{12} \end{bmatrix}_v \quad (3.29)$$

$$\begin{bmatrix} LH_{1,1} \\ HH_{1,1} \end{bmatrix} = T_h \begin{bmatrix} LH_{0,1} \\ HH_{0,1} \end{bmatrix} = T_h \begin{bmatrix} LL_{0,0} & LH_{0,0} \\ HL_{0,0} & HH_{0,0} \end{bmatrix} \begin{bmatrix} T_{21} \\ T_{22} \end{bmatrix}_v \quad (3.30)$$

where (3.29) and (3.30) have made use of (3.25) and (3.26).

For motion compensation with CODWT, prediction from the reference frame is generated from the proper phase in the overcomplete subband(s). If subpixel-accurate motion compensation is used, interpolation can also be performed on the overcomplete subband data. To illustrate, a simple example of the MC process for the HL subband with one level spatial decomposition is depicted in Fig. 3.12. As can be seen from (3.26), (3.27) and (3.29), all subbands are used for generating the reference of the HL subband. The same process can be applied to the LH and HH subbands separately. Alternatively, a triplet of corresponding blocks from the HL, LH, HH subbands (collectively called spatial high (SH) subbands) can be formed so that the same MV is used, saving

Figure 3.12: Formation of the prediction reference for the HL band with CODWT. The subscripts (i, j) , where $i \in \{0, 1\}$ and $j \in \{0, 1\}$, indicate the phase in the horizontal and vertical directions, respectively. The interleaving of four phases may not be needed if the CODWT has properly interleaved the phases in the overcomplete representation.

bitrate for coding MVs. The LL subband should be treated differently than the SH subbands if spatial scalability is desired. In general, the SH bands should not be used for MCTF in the LL subband, or a mismatch between the decoder and encoder would result as the decoder would not have access to those SH subbands.

Since the overcomplete representation generated by OCDWT is equivalent to direct DWT without downsampling, it can be considered as a linear and shift-invariant (LSI) operation. So is the interpolation for MC. Due to the commutative property of LSI operations, the MC (or MCTF) in the overcomplete subbands is equivalent to MC in the spatial domain followed by wavelet transform without downsampling. MC with CODWT is thus at least as efficient as when performed in the spatial domain, in terms of minimizing the energy of the MC residues. 2D+t schemes with such CODWT techniques have been studied and improved coding performance is obtained [19, 51, 52]. On the other hand, the computational complexity for the 2D+t scheme is in general higher than that of t+2D due to the CODWT and ME/MC performed at multiple spatial scales. The number of MVs may also increase. Due to the separation of MC for the SL band and that for the SH bands, the coherence of multiple MVFs may pose a problem.

3.3.3 The 2D+t+2D Scheme

From the discussions in the previous two subsections, spatial decomposition and temporal decomposition are in fact coupled due to motion compensation. In respect to the difficulty in spatial scalability encountered by t+2D schemes and subband MC inefficiency in 2D+t schemes, it motivates a 2D+t+2D scheme that first applies a pre-spatial decomposition to generate a pyramidal representation of the input frames, and then applies MCTF separately to the frames at each spatial scale, followed by a post-spatial decomposition applied to each spatiotemporal subband to exploit residual spatial correlation. Fig. 3.13 shows the overall encoding architecture. Since MCTF is applied at each spatial resolution, it thus avoids the mismatch between the encoder and decoder of t+2D schemes. Since MC is performed at the LL band of spatial transform, it is almost as efficient as in the spatial domain, avoiding the MC inefficiency in 2D+t schemes.

It is obvious that the pyramidal representation after the pre-spatial decomposition is overcom-

Figure 3.13: The overall 2D+t+2D encoding architecture. The block labeled “MUX” selects (and possibly processes) one or more signal to be used for prediction.

plete and redundancies exist across scales. *Inter-scale prediction* (ISP) is thus used to improve coding performance, for which a few possibilities exist [39][53]. In Fig. 3.13, candidate prediction signals are indicated by connections to the multiplexer (labelled “MUX”) that selects one or combines a few to produce the final prediction signal.

3.3.4 Discussion

From the discussions above, it is clear that t+2D schemes favor the full spatial resolution decoding, while 2D+t and 2D+t+2D schemes favor lower resolution decoding as overhead for lower resolution is smaller or none. Unsurprisingly, the choice between t+2D, 2D+t, and 2D+t+2D schemes boils down to an important trade-off: to penalize the highest resolution (2D+t, 2D+t+2D) or to penalize the lowest resolution (t+2D). A comparison between the three schemes is shown in Tab. 3.1, highlighting their relative advantages and disadvantages. In terms of computational complexity, t+2D seems to be the least expensive because ME is performed only at one spatial resolution. As known, ME accounts for a major proportion of total computational complexity, it is therefore desirable to perform ME at minimum spatial resolutions. In this respect, the t+2D scheme is favored over the other two. However, as mentioned above, the spatial scalability for t+2D schemes is problematic due to a mismatch between the decoder and the encoder. We will analyze this problem in detail and propose possible solutions in the next chapter.

Table 3.1: Comparison of three schemes of MCWC

Schemes	Temporal scalability	Spatial scalability	MCTF	Representation redundancy
t+2D	No drift	Drift	Spatial domain, for full resolution only	No
2D+t	No drift	No drift	Wavelet domain, for each resolution	No
2D+t+2D	No drift	No drift	Wavelet domain, for each resolution	Yes

Chapter 4

Spatial Scalability in the t+2D Scheme

As mentioned in Chapter 3, in the t+2D scheme, motion vectors (MV) are obtained only for the full spatial resolution at the encoder and when the decoder reconstructs a lower spatial resolution video, the conventional approach is to apply inverse MCTF (IMCTF) at the target lower spatial resolution with scaled MVs, resulting in a drop in PSNR performance. In this chapter, we present an in-depth analysis on the spatial scalability in the t+2D scheme. In Section 4.1, we study, from a fundamental signal processing perspective of phase, the sub-optimality arising from the scaled motion field used in the conventional decoder and propose a simple method to improve the decoder. As our experiments indicate, though the PSNR performance is greatly improved, it still saturates with the improved decoder. To eliminate the root cause of aliasings that are not cancelable by the decoder, in Section 4.2 we present a detailed anatomy of the MCTF process, and in Section 4.3 propose modifications to the encoder. Throughout the description that leads to the encoder-side solution, the fundamental difference between the t+2D and 2D+t schemes is highlighted. As it turns out, eliminating the aliasing into the lower spatial subbands greatly improves the PSNR performance for the lower spatial resolutions but degrades the PSNR performance for full spatial resolution. Given that the best coding performance is not attainable simultaneously for all spatial resolutions with a fully embedded bitstream, in Section 4.4 we propose a *practical* solution by considering the real world constraints such as transmission cost, storage cost, and computational cost.

4.1 Problems with Scaled Motion Field

In the t+2D scheme, MCTF is applied only once on the full resolution frames. When decoding at a lower spatial resolution, the conventional approach is to apply the inverse MCTF (IMCTF) on the spatially down-scaled frames. Since motion vectors were obtained for full resolution, they need to be scaled according to the spatial reconstruction resolution. For example, when reconstructing at half or quarter resolution, the MVs are divided by two or four, respectively. In the following, we will analyze the decoding operations in detail and show where the problem lies with the conventional

Figure 4.1: Lifting Haar operations

Figure 4.2: The generalized lifting Haar operation

approach.

For clarity, we consider the case of MCTF with lifting Haar transform (see Section 3.2.2 for more details). Fig. 4.1 shows the Haar lifting steps with the lazy transform ignored, where the temporal low-pass frame L is aligned with input frame A and high-pass frame H aligned with input frame B . With this alignment, the motion vectors are estimated from A to B with B as the current frame and A as the reference frame. The high-pass frame H is produced by

$$H = B - P(A) \quad (4.1)$$

and the low-pass frame L produced by

$$L = A + U(H), \quad (4.2)$$

where $P(\cdot)$ denotes the prediction operation and $U(\cdot)$ denotes the update operation.

What the predictor $P(\cdot)$ does is to map A to B by the estimated MVs. Similarly, the updating operator $U(\cdot)$ maps H to A with the reverse MVs. When subpixel-accurate MVs are used, the operators $P(\cdot)$ and $U(\cdot)$ involve interpolating at subsampling positions of A and H , respectively. Due to the similarity of the prediction and update operations, we can generalize them into a uniform representation, as shown in Fig. 4.2, where I indicates *pixel-dependent* interpolation. It is pixel-dependent as the interpolation in both lifting steps is *spatially varying* in general (see Section 3.2 for more details). Note that the generalized representation is also suitable for lifting synthesis operations in IMCTF.

Let us consider the case of one-level spatial scalability for sake of simplicity. Let A_{SL} denote the spatial low-pass (SL) subband and A_{SH} denote the spatial high-pass (SH) subbands collectively of A after wavelet transform with maximal decimation. Similar notations are adopted for B and C .

(a) Processing steps

(b) Steps in (a) rearranged

Figure 4.3: Ideal approach to producing C_{SL}

Without loss of generality, we shall assume that the target frame C is reconstructed from A and B as shown in Fig. 4.2 (with necessary additional information such as MVs). When reconstructing at one level down-scaled resolution, only SL subbands are available at the decoder. That is, in our example, the decoder is to reconstruct C_{SL} only from A_{SL} and B_{SL} . Ideally, C_{SL} should be the result of full frame C undergoing a proper downsampling, i.e., $C_{SL} = F_l(C)$, where F_l is the *reference* low-pass filtering with maximal decimation. As argued in Section 3.3, F_l is taken as the low-pass wavelet filtering used in spatial decomposition by the encoder. Thus, with the notations listed in Tab. 4.1, we have an identity

$$A = LP_S(O(A_{SL})) + HP_S(O(A_{SH})). \quad (4.3)$$

The process of obtaining the reference C_{SL} is depicted in Fig. 4.3(a), where operator I represents

Table 4.1: List of notations

Symbol	Meaning
D	Decimation by 2
O	Interpolation by 2
LP_D	Analysis low-pass wavelet filtering
HP_D	Analysis high-pass wavelet filtering
LP_S	Synthesis low-pass wavelet filtering
HP_S	Synthesis high-pass wavelet filtering

pixel-dependent interpolation according to motion vectors. We can express the reference C_{SL} as

$$C_{SL} = D(LP_D(I(A))) + D(LP_D(B)). \quad (4.4)$$

Since wavelet filtering and interpolation are linear and shift-invariant (LSI) operations, they are commutative with addition. Furthermore, though decimation is shift variant, it is also commutative with addition. Hence, we can rearrange the operations shown in Fig. 4.3(a) into equivalent operations shown in Fig. 4.3(b) that produce the identical result, and accordingly, the reference result may be expressed in an expanded form as

$$\begin{aligned} C_{SL} = & D(LP_D(I(LP_S(O(A_{SL})))))) \\ & + D(LP_D(I(HP_S(O(A_{SH})))))) \\ & + D(LP_D(LP_S(O(B_{SL})))) \\ & + D(LP_D(HP_S(O(B_{SH}))))). \end{aligned} \quad (4.5)$$

4.1.1 Phase Mismatch

The conventional approach to reconstructing C_{SL} is to use A_{SL} , B_{SL} and scaled MVs. For instance, the MV for coefficient (m, n) is that of pixel $(2m, 2n)$ at the full resolution divided by 2. This approach is illustrated in Fig. 4.4(a), where the operator I' represents *coefficient-dependent* interpolation according to scaled MVs. The steps in Fig. 4.4(a) can be expanded into equivalent steps in Fig. 4.4(b), where the chain of interpolation, filtering and decimation fully preserves information. Again, we can rearrange the operations in Fig. 4.4(b) into those in Fig. 4.4(c) with identical result. Note that here it only shows a *mathematical* equivalence, as A_{SH} and B_{SH} are actually unavailable to the decoder. Using the notations adopted earlier, the conventional reconstructed lower spatial resolution frame C_{SL}^c can be expressed as

$$C_{SL}^c = I'(D(LP_D(A))) + D(LP_D(B)). \quad (4.6)$$

Comparing Fig. 4.3(b) and Fig. 4.4(c), it is seen that the lower two branches concerning B are the same, but the upper two branches concerning A are different. The differences lie in the sequence of certain operations and the actual MVs used (original vs. scaled). The difference between C_{SL} and C_{SL}^c is given by

$$\begin{aligned} \Delta^c &= C_{SL}^c - C_{SL} \\ &= I'(D(LP_D(A))) - D(LP_D(I(A))). \end{aligned} \quad (4.7)$$

We investigate the significance of (4.7) under the special case where the motion is purely translational with a global shift of $\boldsymbol{\tau} \equiv [\tau_x, \tau_y]^t$. It then follows that MC interpolation I is an LSI operation and hence the order of I and LP_D can be swapped. Let $A(\boldsymbol{\omega}) \equiv A(\omega_x, \omega_y)$ denote the Fourier transform of A and $H_0(\boldsymbol{\omega})$ denote the Fourier transform of LP_D . Further assume that I is an ideal bandlimited interpolater. Hence $I(A)$ will be identical to A shifted by $\boldsymbol{\tau}$; i.e., $I(A(\boldsymbol{\omega})) = A(\boldsymbol{\omega})e^{-j\boldsymbol{\omega}^t\boldsymbol{\tau}}$. Similarly, assume that the MC interpolator I' in the subband domain is

Figure 4.4: Conventional approach to producing C_{SL}^c

ideally bandlimited so that $I'(A_{SL}(\omega)) = A_{SL}(\omega)e^{-j\omega^t \frac{\tau}{2}}$, where the global shift is scaled down by 2. Then the Fourier transform of Δ^c can be expressed as

$$\begin{aligned}
 \Delta^c(\omega) &= I'(D(H_0(\omega)A(\omega))) - D(H_0(\omega)A(\omega)e^{-j\omega^t \tau}) \\
 &= \frac{1}{4} \sum_{\rho \in \{0,1\}^2} H_0\left(\frac{\omega}{2} + \rho\pi\right) A\left(\frac{\omega}{2} + \rho\pi\right) e^{-j\omega^t \frac{\tau}{2}} \\
 &\quad - \frac{1}{4} \sum_{\rho \in \{0,1\}^2} H_0\left(\frac{\omega}{2} + \rho\pi\right) A\left(\frac{\omega}{2} + \rho\pi\right) e^{-j(\frac{\omega}{2} + \rho\pi)^t \tau} \\
 &= \frac{1}{4} \sum_{\rho \in \{0,1\}^2 \setminus (0,0)} H_0\left(\frac{\omega}{2} + \rho\pi\right) A\left(\frac{\omega}{2} + \rho\pi\right) (1 - e^{-j\pi \rho^t \tau}) e^{-j\omega^t \frac{\tau}{2}} \quad (4.8)
 \end{aligned}$$

where phase $(0,0)$ has been canceled out. It is interesting to observe the result of (4.8) for a few special cases of τ . When $\tau = [2n, 2m]^t$ for any integers n and m , i.e., a shift of an even number of samples in both directions, we have $\Delta^c(\omega) = 0$. When $\tau = [2n+1, 2m+1]^t$, i.e., a shift of

Figure 4.5: Proposed approach to producing C_{SL}

an odd number of samples in both directions, phase $(1, 1)$ cancels out and phases $(0, 1)$ and $(1, 0)$ survive. When $\tau = [2n, 2m + 1]^t$ or $\tau = [2n + 1, 2m]^t$, phase $(1, 1)$ survives, and phase $(1, 0)$ or $(0, 1)$ cancels out. In a more general case of non-integer shift with $\tau_x + \tau_y \neq 2n$, all three phases shall survive. This clearly shows that there is a *phase mismatch* between the decoder (that follows the conventional approach) and the encoder (that the ideal approach follows) in lower spatial resolution reconstruction. Since all the surviving terms are spatial high frequencies for at least one direction and H_0 is a low-pass filter, the final total energy is limited.

We would like to highlight that the analysis presented here is different from that in [54] due to one different fundamental assumption. We have assumed that the reference lower spatial resolution sequence is formed by the LL subbands obtained during the spatial decomposition at the encoder, whereas in [54] the reference is the LL subbands resulted from *ideal bandlimited* spatial analysis, which excludes the frequency leakage from the SH subbands.

4.1.2 Improved Decoding with Up-scaling

Following the discussion on the phase mismatch problem in the previous section, it is reasonable to expect a reduction in the difference given by (4.7) by aligning the MC interpolation phases at the decoder with that at the encoder when performing IMCTF for lower spatial resolution decoding. Specifically, we propose to first *up-scale* the lower spatial resolution frames to full spatial resolution, apply IMCTF with original MVs, and then down-scale the reconstructed (full resolution) frames to the target resolution. Through this way, the decoder exactly mimics the encoder except that some spatial high bands are in fact nullified (filled with zeros) during 2D-IDWT. Fig. 4.5(a) illustrates the steps of the proposed approach, which can be equivalently represented as in Fig. 4.5(b). Comparing Fig. 4.5(b) and Fig. 4.3(b), it can be seen that all operations on A_{SL} and B_{SL} are the same, and the only difference is that A_{SH} and B_{SH} are unavailable to the decoder. The result of the proposed method, denoted C_{SL}^p , can be expressed as

$$C_{SL}^p = D(LP_D(I(LP_S(O(A_{SL})))))) + D(LP_D(LP_S(O(B_{SL}))))). \quad (4.9)$$

The difference between the proposed C_{SL}^p to the ideal C_{SL} (refer to the expanded form in (4.5)) is given by

$$\begin{aligned}\Delta^p &= C_{SL}^p - C_{SL} \\ &= D(LP_D(I(HP_S(O(A_{SH})))))) + D(LP_D(HP_S(O(B_{SH})))) \\ &= D(LP_D(I(HP_S(O(A_{SH}))))),\end{aligned}\tag{4.10}$$

where the last step is due to the fact that $D(LP_D(HP_S(O(B_{SH}))))$ is equal to zero for any PR subband transform.

To investigate the characteristics of Δ^p , again we look at its Fourier transform by assuming that motion is perfectly modeled as a global shift and that the I is an ideal bandlimited interpolator, same assumptions under which (4.8) is obtained. We denote $G_1(\omega)$ as the Fourier transform of HP_S . We have

$$\Delta^p(\omega) = \frac{1}{4} \sum_{\rho \in \{0,1\}^2} H_0\left(\frac{\omega}{2} + \rho\pi\right) G_1\left(\frac{\omega}{2} + \rho\pi\right) A_{SH}(\omega) e^{-j\left(\frac{\omega}{2} + \rho\pi\right)^t \tau} \tag{4.11}$$

It can be shown that under the special case of $\tau = (2n, 2m)$, $\Delta^p(\omega) = 0$ for any PR subband transform. It is also observed that A_{SH} is subjected to both low-pass filtering and high-pass filtering and the final distortion is thus attenuated. Consequently, for the proposed method, wavelet filters with better cutoff performance can further reduce distortion when a decoding lower spatial resolution video. In contrast, the distortion arising from the conventional method will always contain some energy even when ideal brick-wall wavelet filters are used (see (4.8)).

It only recently came to the authors' knowledge that Xiong et al. proposed the same solution in [55] where they offered arguments mainly from an operational point of view. Though the same end solution is independently proposed in the thesis, we have offered and based our solution on an alternative analysis on the sub-optimality of the conventional approach to decoding a lower spatial resolution video.

4.1.3 Experimental Results and Discussion

To test the proposed method of up-scaling, two experiments have been carried out with the MC-EZBC coder [17] that assumes the t+2D scheme. The code of MC-EZBC was downloaded from the MPEG CVS server on 20 December, 2005. Six CIF sequences downloaded from the Hannover FTP server [56], namely *Foreman*, *Bus*, *Mobile*, *Stefan*, *Flower garden* and *Coastguard* are tested. They are in raw YUV 4:2:0, progressive format.

Experiment 1 tests the PSNR performance for decoding at QCIF resolution. The conventional approach performs IMCTF directly on the spatially reconstructed QCIF frames with down-scaled MVs, while the proposed method first up-scales the spatially reconstructed QCIF frames to CIF resolution, then performs IMCTF on these CIF frames and at last down-scales the reconstructed frames to QCIF resolution.

Experiment 2 tests the PSNR performance for decoding at QQCIF resolution. Again, the conventional approach performs IMCTF directly on the spatially reconstructed QQCIF frames with

down-scaled MVs. We test two cases with the proposed method. In the first case, the proposed method scales QQCIF frames one level up to QCIF resolution; and in the second case, the proposed method scales QQCIF frames two levels up to the original CIF resolution.

The experimental results are shown in Fig. 4.6 and Fig. 4.7. As can be seen, the proposed up-scaling method consistently outperforms the conventional method and the improvement is larger at higher bitrates. For QCIF decoding, the improvement is about 0.8 dB to 1.1 dB at 750 kbps. For QQCIF decoding, the improvement is about 1.0 dB to 1.3 dB (except for *Stefan*, which is 0.6 dB) at 250 kbps. It is also interesting to note that in Experiment 2, one-level up-scaling performs almost as well as two-level up-scaling. This may be due to the aliasing in the SL band being dominant in the total distortion, which cannot be cancelled without the SH bands (a topic of the remainder of this chapter). On the other hand, we also note that the PSNR performance still saturates at very high rates, which is undesirable.

To summarize, the advantages of the proposed method are mainly two-fold:

- No change to the encoder, no additional information needs to be transmitted.
- It allows for very flexible implementation. It can be made an optional mode for the decoder, which can be enabled or disabled virtually at any time. For instance, when complexity becomes an issue, it can be enabled for one GOP and disabled for the next so as to meet the total computation constraints.

The disadvantage, on the other hand, is of course the increased computational complexity for the decoder. This is very much implementation-dependent and it may be partially alleviated by a control mechanism that can switch on and off the up-scaling method depending on the available resources.

4.2 Anatomy of the MCTF Process

From (4.5), (4.9) and (4.10), it is obvious that the proposed decoding method can be further improved by minimizing the contribution from A_{SH} and/or that from B_{SH} at the encoder. As mentioned, one possible way is to use wavelet filters with better cutoff performance. Alternatively, we may exclude A_{SH} and/or B_{SH} from contributing to C_{SL} , so that no distortion would occur during reconstruction. This means that A_{SH} and/or B_{SH} should not be used in the lifting step. Note that here A and B are the inputs to the generalized lifting step shown in Fig. 4.2; they represent different frames for the prediction step and for the update step. In this section, we provide a detailed anatomy of the MCTF process at the encoder and see how the contribution from A_{SH} and/or B_{SH} to C_{SL} can be most effectively eliminated with minimal impact on overall R-D performance.

We start with a detailed anatomy of the lifting step. For simplicity, we consider one level scalability. Recall that both the prediction step and the update step can be represented by a uniform structure as shown in Fig. 4.2, which can be redrawn as in Fig. 4.8(a), where the intermediate motion compensated frame (by linear and shift-variant (LSV) interpolation) is denoted as A' . Let A_L denote a full resolution frame obtained by inverse transforming a wavelet domain frame with A_{SL} as the

Figure 4.6: PSNR performance of proposed decoder vs. conventional decoding. CIF sequences are decoded at QCIF resolution and 30 fps.

Figure 4.7: PSNR performance of proposed decoder vs. conventional decoding. CIF sequences are decoded at QQCIF resolution and 30 fps.

LL band and with the SH (i.e., HL, LH, and HH) bands zeroed. Similarly, let A_H denote a full resolution frame obtained by inverse transforming a wavelet domain frame with the LL band zeroed and with A_{SH} as the SH bands. Apparently, A_L and A_H are the spatial low-pass component and high-pass component of A , respectively, and $A = A_L + A_H$. Thus, the MC interpolation applied to A can be applied to A_L and A_H separately, resulting in MC interpolated frames A'_L and A'_H , respectively. Fig. 4.8(b) illustrates the process of obtaining A'_L and A'_H . It is easy to see that $A' = A'_L + A'_H$, and the lifting output can be equivalently obtained by

$$C = B - A'_L - A'_H. \quad (4.12)$$

Let $A'_{L_{SL}}$ and $A'_{H_{SL}}$ denote the LL band of the (intermediate) frames A'_L and A'_H , respectively. Similarly, let $A'_{L_{SH}}$ and $A'_{H_{SH}}$ denote the SH bands of the (intermediate) frames A'_L and A'_H , respectively. Since A'_H is obtained from MC interpolation on A_H , the spatial high-pass component of A , $A'_{H_{SL}}$ is the energy (aliasing) scrambling from the SH bands of A (or A_H) into the LL band of A'_H (or A') due to motion compensation. Similarly, $A'_{L_{SH}}$ is the energy (aliasing) scrambling from the LL band of A (or A_L) into the SH bands of A'_L (or A').

According to (4.12), the LL band C_{SL} and SH bands C_{SH} of the lifted frame C are given by

$$C_{SL} = B_{SL} - A'_{L_{SL}} - A'_{H_{SL}}, \quad (4.13)$$

and

$$C_{SH} = B_{SH} - A'_{L_{SH}} - A'_{H_{SH}}, \quad (4.14)$$

respectively. The breakdown of the lifting output is depicted in Fig. 4.8(c), where the origin of each element is clearly traceable.

From (4.13), it is trivial to see that B_{SL} can be perfectly reconstructed by

$$B_{SL} = C_{SL} + A'_{L_{SL}} + A'_{H_{SL}}. \quad (4.15)$$

When targeting at half resolution, the SH bands are unavailable at the decoder, hence the decoder reconstructs B_{SL} by

$$B_{SL} = C_{SL} + A'_{L_{SL}}. \quad (4.16)$$

Clearly, without A_H , the aliasing $A'_{H_{SL}}$ can not be cancelled out at the decoder, unless it is zero. In the following section, we will show that the aliasing $A'_{H_{SL}}$ is in general not zero due to the *subband scrambling* caused by motion compensation.

(a) The generalized MC lifting step

(b) Anatomy of the MC interpolation on A

(c) Anatomy of the lifting output

Figure 4.8: The detailed breakdown of the lifting step. The dashed subbands in (a) are filled with zeros.

4.2.1 Subband Scrambling

With reference to Fig. 4.8(a), consider the lifting prediction for generating the high-pass frame H . Pixels in A are mapped to corresponding positions determined by the estimated MVs, and then the mapped A , denoted as A' , is subtracted from B to produce H . This mapping of A can be thought of as selecting pixels from A and putting them in an arbitrary order. Due to the sensitivity of wavelet coefficients to the location in the spatial domain, A' would have a different spectrum (in the wavelet domain) than A because some frequencies may be created due to the new vicinities. These new frequencies in the SL band are not necessarily only due to the original low frequencies; they may be created from some high frequencies. Hence, we say that energy *scrambles* from the SH bands into the SL band. Similarly, there is also energy scrambling from the SL band into the SH bands. Consequently, the spectrum of H will contain such scrambled energy. Similar arguments hold for the update step.

To illustrate the subband scrambling problem with a real example, we conducted a simple experiment with the first two frames from the *Foreman* CIF sequence. We name the first frame A and the second frame B . The steps are summarized as follows:

1. Perform one level of wavelet transform with biorthogonal 9/7 wavelets on A , resulting in four bands, namely LL, HL, LH, and HH. Compose a wavelet domain image A_L^w with band LL and nullified (set to zeros) bands HL, LH, and HH. Then inverse transform A_L^w to the spatial domain to produce an image A_L that only contains low frequencies. Similarly, compose a wavelet domain image A_H^w with bands HL, LH and HH and with band LL nullified, then inverse transform it to produce an image A_H that only contains high frequencies.
2. Repeat Step 1 on frame B , obtaining spatial low-pass frame B_L and high-pass frame B_H .
3. Do motion estimation with B as the current frame and A as the reference frame.
4. Do lifting Haar MCTF on (A_L, B_L) to generate temporal low-pass frame L_L and temporal high-pass frame H_L , with the motion vectors obtained in Step 3.
5. Repeat Step 4 on (A_H, B_H) , obtaining in temporal high-pass frame L_H and low-pass frame H_H .
6. Do one level of wavelet transform on L_L, H_L, L_H , and H_H , resulting in L_L^w, H_L^w, L_H^w , and H_H^w , respectively.

■

The spatial low-pass frames A_L^w, B_L^w, L_L^w , and H_L^w are shown in Fig. 4.9, and spatial high-pass frames A_H^w, B_H^w, L_H^w , and H_H^w in Fig. 4.10. It is clearly seen from Fig. 4.9 that after MCTF, some high frequencies appear in the SH bands, which were nullified (zeroed) before MCTF. Similar observation can be made on the spatial high-pass frames in Fig. 4.10. This verifies the *subband scrambling* problem.

Remark: The subband scrambling discussed here is different from the aliasing, called *energy spillover*, caused by downsampling with non-ideal filters in a multi-rate system. Subband scrambling

Figure 4.9: Subband leakage illustration: wavelet data of spatial low-pass frames before and after MCTF. For better visualization, the dynamic range is first amplified 16x and then clipped to within $[0 \ 255]$.

Figure 4.10: Subband leakage illustration: wavelet data of spatial high-pass frames before and after MCTF. For better visualization, the dynamic range is first amplified 16x and then clipped to within [0 255].

is caused by the MC in the temporal dimension and it occurs on top of the energy spillover that is introduced during spatial decomposition after MCTF.

4.3 Encoder Side Solution

As verified in the previous section, the aliasing signal A'_{H_SL} into the SL band of the lifting output is non-zero in general. As a result, when decoding a lower spatial resolution video, the decoder will not be able to cancel out A'_{H_SL} because it can only be reproduced from A_H and A_H is unavailable. This leads to two options to eliminate the aliasing from SH bands into SL band.

- **Option 1** is to exclude the aliasing signal A'_{H_SL} from the lifting step at the encoder such that

$$C_{SL} = B_{SL} - A'_{L_SL}, \quad (4.17)$$

and consequently, the decoder can perfectly reconstruct B_{SL} using (4.16).

- **Option 2** is to transmit the aliasing signal A'_{H_SL} when targeting at the lower resolution, so that the decoder can apply (4.15) to perfectly reconstructing B_{SL} .

Let us first check the feasibility of option 2. First of all, let us point out that aliasing is incurred in both lifting steps, which means there is one aliasing signal for the prediction step and one aliasing signal for the update step. This is also true for the *intermediate* frames during multiple-level MCTF, as the aliasing incurred in these intermediate frames is also needed by the decoder to perfectly reconstruct higher temporal frames (refer to Fig. 3.2). If the GOP size is N and T levels of MCTF are applied, $N = a2^T$ for some positive integer a , then there will be a total of $N(1 + \frac{1}{2} + \dots + \frac{1}{2^{T-1}})$ aliasing signals. Second, note that there is no aliasing signal for the full spatial resolution bitstream, but the bitstreams for lower spatial resolutions all include some aliasing signals that are of the target spatial resolution. For example, the aliasing A'_{H_SL} overlaps with C_{SL} for the half resolution video as shown in Fig. 4.8. Given the significant size of the aliasing signals, for better R-D performance it is certainly desirable to use as few aliasing signals as possible while maintaining perfect reconstruction. In this respect, the aliasing signal in the update step may be excluded from the updated output as in option 1, because the update step has relatively less impact on the overall R-D performance than the prediction step. In this case, only the aliasing signals associated with temporal high-pass frames are included in the bitstream, and there will be only $N(\frac{1}{2} + \frac{1}{4} + \dots + \frac{1}{2^T})$ aliasing signals, half of the previous case. However, even with this simplification, the redundancy in the lower spatial resolution bitstreams will still severely affect the R-D performance, compared to the case of directly coding the target resolution frames by the original coder. Hence, we forgo this option and instead concentrate on the first option.

The key idea suggested by option 1 is that spatial bands belonging to a lower resolution can be used in the lifting of a band belonging to a higher resolution, but the reverse is not allowed so as to make each spatial scale self-contained. Hence, we name option 1 as a *low-to-high lifting MCTF* (LTH-MCTF) approach. From Figs. 4.8(b) and (c), we can easily derive the steps to generate

With the proposed LTH-MCTF approach, the spatial band corresponding to the lowest supported spatial resolution is MCTFed only with information in that band. In other words, ignoring the actual scale where MCTF is performed, these bands are MCTFed as if they were the input frames. Hence the coding performance of the LTH-MCTF approach is readily optimum for the lowest spatial resolution in the sense that no better MC prediction can be done. However, for higher spatial resolutions, the bands in HL, LH and HH directions do not utilize the ‘full’ information in MCTF, where the ‘full’ information means all of the bands included in the target resolution. Hence, the coding performance may not be optimal, as compared to when ‘full’ information is used in MCTF for every band.

Since the proposed LTH-MCTF approach performs MCTF at the full resolution, when decoding a lower spatial resolution video, the previous argument in Section 4.1.2 applies. Hence, at the decoder perfect reconstruction can be obtained by following the LTH-MCTF approach to eliminate the source of aliasing at the encoder and following the up-scaling method to match the decoder’s behavior to that of the encoder.

4.3.1 Fundamental Difference between $t+2D$ and $2D+t$ with Complete-to-Overcomplete DWT (CODWT)

It is interesting to note that, as far as the source of the prediction/update signal is concerned, the $2D+t$ scheme with Complete-to-Overcomplete DWT(CODWT) may be configured to behave the same way as the proposed LTH-MCTF approach. The decomposition steps of the $2D+t$ scheme with CODWT for one such configuration are shown in Fig. 4.12(a), where the source of the prediction/update signal for each spatial scale only comes from the same and lower scales (if any). The detailed breakdown of the lifting step for one scale is depicted in Fig. 4.12(b) as an example. The difference between the proposed LTH-MCTF approach and the depicted $2D+t$ scheme with CODWT lies in the domain where MCTF is performed: spatial domain for the proposed solution, and wavelet domain for the $2D+t$ scheme with CODWT.

With reference to Fig. 4.11, if not nullifying the dashed bands, the generated prediction/update signal is in fact the same as that generated by the original $t+2D$ scheme. In this case, all spatial bands are used to predict/update any spatial band. Such a flexibility in MCTF lifting operation can also be realized in the $2D+t$ scheme with CODWT in which spatial scales are not maintained to be self-contained, as shown in Fig. 4.13(a). For clarity, we also show the detailed breakdown of the lifting step for one scale in Fig. 4.12(b) as an example.

From the above description, it now should be clear that the fundamental difference between the $t+2D$ scheme and the $2D+t$ scheme is what spatial bands are used to predict/update one spatial band in MCTF, rather than the order of performing the spatial and temporal decomposition. As long as the source of prediction/update signals is well controlled, we can obtain an equivalent realization of one scheme with the other (of course, minor differences exist because the MCTF is performed in different domains).

4.3.2 Experimental Results and Discussion

Experiment has been conducted to test the performance of the proposed LTH-MCTF approach. Again, the MC-EZBC coder was used. In this experiment, we chose to support two-level scalability, so that we compare the performance for full, half and quarter resolutions. For the original encoder, we used the proposed decoder to decode half and quarter resolution bitstreams as it is known to perform better than the original decoder. For the LTH-MCTF encoder, we tested with both the original decoder and the proposed decoder for half and quarter resolution bitstreams. The PSNR performance for the Y component of two sequences, *Foreman* and *Mobile*, are shown in Figs. 4.14, 4.15 and 4.16.

It is easily seen that for full resolution, LTH-MCTF experiences great PSNR loss compared to the original encoder, as much as 1.9 dB for *Foreman* and 4 dB for *Mobile* at high bitrate. It is also interesting to note that the more spatial details a sequence has, the greater the performance penalty is. For half resolution, LTH-MCTF with the proposed decoder outperforms the original encoder when bitrate is greater than 500 kbps. However, when bitrate is less than 500 kbps, surprisingly, the original encoder performs better. It could be due to that at low bitrates, the quantization noise causes larger distortion than the scrambled aliasing does. It is also observed that even with the

(a) Decomposition steps

(b) Breakdown of the lifting step for one spatial scale (the middle row in (a))

Figure 4.12: The 2D+t scheme with CODWT where spatial scales are self-contained. In (a), the arrows show the source of the prediction/update signals. In (b), spatial bands from both the target and the lower scales are used to produce the final prediction/update signal.

(a) Decomposition steps

(b) Breakdown of the lifting step for one spatial scale (the middle row in (a))

Figure 4.13: The 2D+t scheme with CODWT where spatial scales are not self-contained. In (a), the arrows show the source of the prediction/update signals. In (b), all spatial bands are used to produce the final prediction/update signal.

(a)

(b)

Figure 4.14: PSNR performance of the proposed LTH-MCTF encoder for full resolution.

(a)

(b)

Figure 4.15: PSNR performance of the proposed LTH-MCTF encoder with and without proposed decoder for half resolution.

(a)

(b)

Figure 4.16: PSNR performance of the proposed LTH-MCTF encoder with and without proposed decoder for quarter resolution.

original decoder, LTH-MCTF still outperforms the original encoder with the proposed decoder at medium to high bitrates, though the turning point is slightly higher than when proposed decoder is used. This indicates that the scrambled aliasing causes more distortion than the phase mismatch between the decoder and encoder. More importantly, the PSNR performance of LTH-MCTF does not saturate and it increases as bitrate increases. This is in sharp contrast to the original encoder whose PSNR performance saturates at around 1000 kbps. Even at 1000 kbps, LTH-MCTF already outperforms the original encoder by about 3 dB for *Foreman* and 4 dB for *Mobile*. In general, similar observations can be made for quarter resolution.

In summary, LTH-MCTF performs the better for lower resolutions at medium to high bitrates, but it has significant performance loss for full resolution. Essentially, LTH-MCTF and the original t+2D encoder represent two extremes: optimal for the lowest resolution, or optimal for the full resolution. In fact, it is also possible to design schemes in-between these extremes that are optimal for a medium resolution. It simply involves excluding SH bands above the target resolution from the MC prediction/update of bands below or belonging to the target resolution, so that the target resolution is self-contained but the lower resolutions allow scrambled aliasings. Once a target resolution is chosen to be optimized, the other resolutions will all be penalized, either due to the inefficiency of MC prediction for those above the target resolution, or due to scrambled aliasings not cancelable by the decoder for those below the target resolution. Thus, the best R-D performance is not attainable simultaneously for all spatial resolutions.

4.4 One Practical Solution: The Pyramidal t+2D Scheme

Having established that the best R-D performance for all spatial resolutions is not attainable simultaneously with a fully embedded bitstream, it is worthwhile to study a *practical* solution instead. In defining the practicality of a solution, we mainly consider the major constraints in an end-to-end video system: computational cost, storage cost and transmission cost. The computational cost is associated with both the encoder and the decoder, whereas the storage cost is mainly associated with the encoder as the decoder just receives what is delivered. The transmission cost may be measured by the increase in bitrate per one dB improvement in PSNR (i.e., kbps/dB), which may in turn translate into monetary costs of the system. In this thesis, we do not attempt to quantify these costs and only qualify them. As per today's state of the art, transmission cost is the most important factor when designing an end-to-end video system, as the price of computation power has been dropping much more quickly than that of transmission. The price of storage has been dropping even faster. Consequently, we may sacrifice storage at the encoder for reducing the transmission cost. This means that the R-D performance of the transmitted bitstream is to be optimized. Such a compromise can be achieved by simply generating one independently R-D optimized bitstream for each spatial resolution, and transmitting the corresponding bitstream when targeting a particular resolution. Specifically, 2D-DWT is first applied to the input frames to generate an image pyramid for each frame, then the frames belonging to the same pyramid level (i.e., the same spatial resolution) are coded by the original t+2D scheme. The overall coding architecture is depicted in Fig. 4.17, and we call it a *pyramidal t+2D* scheme. Apparently, it resembles a 2D+t+2D structure described

Figure 4.17: The proposed pyramidal t+2D scheme. Note that after the pre-spatial decomposition, the frames in each scale undergo t+2D coding separately and result in independent bitstreams (possibly with some common data, e.g., motion vectors).

in Section 3.3.3, but does not use inter-scale-prediction (ISP). Note that the final bitstream is an overcomplete representation of the input and thus not fully embedded.

Let us look at the impact on the computation and storage. Assume S -level spatial scalability is supported. Also assume the motion estimation is only done at the full resolution and the found motion vectors are used for all scales (with proper scaling). The total data to be coded, expressed in proportion of the original data, is $1 + \frac{1}{4} + \frac{1}{16} + \dots + \frac{1}{2^{S-1}}$, which is upper bounded by $1\frac{1}{3}$. If we assume the compressibility of frames at different spatial levels are relatively constant, the final complete bitstream is at most about $1\frac{1}{3}$ times that for the full resolution, representing a 33% increase in stored size. For storage purpose, this increase in size may be considered acceptable given storage's ever increasing capacity and decreasing price today. In terms of computation, for the encoder it is quite comparable to the original t+2D scheme if motion estimation is conducted only once at the full resolution, and it is slightly lower than that of the full-fledged 2D+t+2D scheme, because ISP is not performed. The complexity for the corresponding decoder is in fact lower than that of the proposed decoder, as IMCTF is performed at the target resolution directly.

4.4.1 Experimental Results and Discussion

Experiments have been conducted to test the performance of the proposed pyramidal t+2D scheme. Again, the MC-EZBC coder was used, and two-level scalability is supported. The PSNR performance for the Y component of two sequences, Foreman and Mobile, are shown in 4.18 and

4.19, where the R-D curves for the original t+2D scheme and the LTH-MCTF approach are also included for comparison.

For full spatial resolution, since the proposed pyramidal t+2D scheme produces exactly the same bitstream as the original t+2D scheme does, the PSNR performance is also the same as that shown in Fig. 4.15. For half resolution, the proposed pyramidal t+2D scheme consistently outperforms the original t+2D scheme and the proposed LTH-MCTF approach. The improvement over LTH-MCTF is about 0.7 dB for Foreman and 2 dB for Mobile at high bitrates. For quarter resolution, the pyramidal t+2D scheme and LTH-MCTF perform rather closely. This is not surprising, as the performance for the lowest resolution is optimized in LTH-MCTF. The difference between the pyramidal t+2D scheme and LTH-MCTF in this case lies in the scale where MCTF is performed: quarter resolution frames for the pyramidal t+2D scheme, and full resolution frames for LTH-MCTF. In summary, optimal performance for each spatial resolution is attainable with the proposed pyramidal t+2D scheme, at the cost of extra storage at the encoder.

(a)

(b)

Figure 4.18: PSNR performance of the proposed pyramidal t+2D scheme for half resolution.

(a)

(b)

Figure 4.19: PSNR performance of the proposed pyramidal $t+2D$ scheme for quarter resolution.

Chapter 5

Set-partitioning in MCWC

It is found in [57] in the context of image coding that a simple partitioning of wavelet coefficients into two subsets, *significant* and *insignificant*, with respect to a threshold, can reduce the total entropy¹ and thus can lead to improved coding performance. The success of this partitioning on natural images is due to the energy clustering property of wavelet transform around image textures and the efficiency of partitioning by morphological dilation. This motivates us to study the statistical properties of the *temporal high-pass frames* produced by MCTF that do *not* resemble natural images, and to investigate possible set-partitioning techniques for improving coding performance. Indeed, it is observed that with variable size block motion estimation/compensation, the motion-compensated prediction residues exhibit different properties in motion blocks of different sizes. Hence it seems possible to exploit the motion block partitioning for statistical contextual modeling. In this chapter, we first review the basics of variable size block motion estimation, and validate the hypothesis that different statistical properties exist in motion blocks of different sizes. We then propose a simple method of set-partitioning aligned with motion blocks in Section 5.2. In the last section of this chapter, we conduct experiments to estimate the reduction in entropy and to test if the reduction can turn into actual improvement in PSNR performance.

5.1 Variable Size Block Motion Compensation

In block-based motion compensation (MC), it is well known that small motion blocks result in smaller motion-compensated prediction residues than large motion blocks. This can be intuitively reasoned by considering the large blocks as a composition of smaller ones, hence MC with these sub-blocks is guaranteed to have the identical residues if they all use the same motion vectors (MV). By further searching, it is possible to place small motion blocks in more optimal positions so that total residues are reduced. However, small motion block sizes lead to more MVs, which in turn lead to more bits to represent these MVs. Hence a trade-off between the motion block size and

¹The term “entropy”, in the context of coding, in fact refers to the lower bitrate bound of a coding process driven by a particular set of probability estimates, which is generally affected by the way the probability estimates are obtained (e.g. the transformations involved in the coding process).

MC residues is necessary, which relates to the bit budget trade-off between sample data and motion data. As a result, researchers have developed variable size block motion estimation/compensation schemes to address this problem. For instance, tree-structured macroblock partitions are used for motion compensation in H.264 [3][58][59] with lots of fast algorithms for deciding the mode developed [60][61], and *hierarchical variable size block motion estimation/compensation* (HVSMBE/HVSBMC) is used in 3D wavelet coders MC3D-FSSQ [15], MC-EZBC [17] and VidWav [62]. As variable size motion blocks can be conveniently organized in a hierarchical tree-structure, in the sequel we will use the term hierarchical block motion estimation/compensation and the term variable size block motion estimation/compensation (VSBME/VSBMC) interchangeably.

Hierarchical motion estimation consists of two main processes, a motion block splitting processing for building the initial MV tree [63] and an MV tree-pruning (motion block merging) process for finding the rate-distortion (R-D) optimized MV tree [64]. The initial MV tree is built in top-down order: start with large motion blocks and search for optimal motion vectors; divide the large block into smaller sub-blocks and search for optimal motion vectors for each of them; compare the distortion measure (e.g., *sum of absolute difference* (SAD), *mean squared error* (MSE)) resulted from the large block with the total from the sub-blocks; and finally based on some predefined criteria a decision is made as whether to divide the large block, and if so this process is recursively applied to each of the sub-blocks. An example is shown in Fig. 5.1 to illustrate an example the HVSMBE splitting process and its corresponding MV tree. It should be noted that hierarchical motion estimation in general does not require the block shape to be square; other shapes such as rectangles may be used, as in H.264.

A simple criterion for deciding the splitting of a motion block is described here to facilitate discussions that follow. Let B represent a motion block and $B^i, i = 1, 2, \dots, n$, represent a sub-block, so that the union of B^i 's exactly covers B with no overlapping. Let (B_{dx}, B_{dy}) be the motion vector found for block B , and $D(B_{dx}, B_{dy})$ the associated distortion. Similarly, let (B_{dx}^i, B_{dy}^i) denote the found motion vector for sub-block B^i and $D(B_{dx}^i, B_{dy}^i)$ the associated distortion. Then block B is divided into sub-blocks if the prescribed minimum block size has not been reached and the following condition is true:

$$D(B_{dx}, B_{dy}) > a \sum_{i=1,2,\dots,n} D(B_{dx}^i, B_{dy}^i) \quad (5.1)$$

where parameter a controls the trade-off between the distortion and the number of motion vectors, and $a = 1$ in the simplest case.

The initial MV tree is then optimally pruned subject to a R-D constraint. This is not a simple one-dimensional optimization problem; rather it involves constrained optimization of each MV field and joint optimal rate allocation of MV fields and 3-D subbands [15]:

$$\operatorname{argmin}_{(R_{mv}, R_{3D})} D_{cod} \in \{(R_{mv}, R_{3D}) | R_{mv} + R_{3D} \leq R_T\} \quad (5.2)$$

where R_{mv} is the rate of MVs, R_{3D} is the rate of 3-D subbands, and R_T and D_{cod} are the total rate and distortion. The optimal solution can be found by simultaneously adjusting the bit allocation to MVs and subbands. Fig. 5.2 illustrates the interaction between R_{mv} and R_{3D} with respect to the

(a) Splitting process

(b) MV tree representation

Figure 5.1: Example of motion block partitioning in hierarchical motion estimation. (a) The original block is divided into four sub-blocks, two of which are further divided into four sub-blocks each. (b) The corresponding MV tree, where the shaded boxes are the final motion blocks.

total distortion, subject to a given total bitrate. A different curve exists for a different total bitrate and the solution to (5.2) is a pair of (R_{mv}, R_{3D}) in a two-dimensional space that gives the minimum total distortion, representing an optimal trade-off between MVs and subband data.

For a given R_{mv} , the optimal MV tree is the one that minimizes the final distortion. However, the final distortion according to a given MV tree is not readily available until 3-D subbands are coded. Fortunately, the variance of the MC prediction residues serves as a good approximation to the final distortion, which is widely used as the objective measure in MV tree pruning. With the found optimal MV tree, the 3-D subbands are generated and then quantized and coded, subject to the remaining bitrate budget. In the case of embedded 3-D subband coding, where bitplane coding is employed, there is no fixed quantization size and the bitrate budget effectively increases successively over bitplanes. Thus, it would be more precise to formulate the joint optimization problem as this: to achieve a certain distortion level (i.e., coding up to a certain bitplane), find the minimum total bitrate needed for MVs and 3-D subbands. Note that this formulation still involves interaction between MVs and subbands. Clearly, at the time of encoding, there is no way to know the bitrate budget and joint optimization is thus impossible. In the sequel, we describe the common approach to optimizing the MV tree independently of 3-D subbands in hopes of gaining insight into

Figure 5.2: Illustration of the interaction between R_{mv} and R_{3D} on the total distortion. The horizontal line is given by $R_T = R_{mv} + R_{3D}$. Point A corresponds to a fully preserved MV tree. Point C corresponds to the other extreme case where no MC prediction is done. Point B corresponds to an optimal rate allocation to R_{mv} and R_{3D} where total distortion is minimized. By changing R_T , a different curve is obtained.

the relationship between motion block and MC residual energy.

The constrained optimization (i.e., optimal pruning of MV tree) of the MV filed is formulated using Lagrange Multipliers as [65]

$$J_{mv} = D_{mv} + \lambda_{mv} R_{mv}, \quad (5.3)$$

where D_{mv} is the variance of MC prediction residues and λ_{mv} is a Lagrangian parameter. The goal is to minimize J_{mv} . Note that R_{mv} can be modified by varying the MV tree. If λ_{mv} is given, J_{mv} is minimized at the point of the R-D curve where the slope is λ_{mv} , as depicted in Fig. 5.3; that is, $\frac{\Delta D_{mv}}{\Delta R_{mv}} = \lambda_{mv}$. A very large value for λ_{mv} corresponds to a maximally pruned MV tree, and a small value to a maximally preserved one. Typically, a small value is used for λ_{mv} in MV tree pruning, and this inevitably compromises the R-D performance for low bitrates because a large percentage of bitrate is used by MV coding. This drawback can be mitigated by coding the MVs in a scalable manner [34].

We reckon that there is a relationship between the motion block size and the MC prediction residual energy, by assuming that the MV tree is pruned with a given λ_{mv} . Let t denote a node of the MV tree and S_t denote all its descendant leaf nodes. Then $D(t)$ shall denote the distortion if the node t is a leaf node, and $R(t)$ shall denote the bitrate needed for transmitting the MV for node t . Similarly, $D(S_t)$ and $R(S_t)$ denote respectively the distortion and MV bitrate if node t is an intermediate node. Then the R-D slope for node t is given by

Figure 5.3: The rate-distortion curve. The magnitude of the slope is monotonically decreasing as bitrate increases.

$$\begin{aligned}
 \frac{\Delta D(S_t)}{\Delta R(S_t)} &= \frac{D(t) - D(S_t)}{R(S_t) - R(t)} \\
 &= \frac{D(t) - \sum_{\tau \in S_t} D(\tau)}{\sum_{\tau \in S_t} R(\tau) - R(t)}
 \end{aligned} \tag{5.4}$$

Thus, the descendants of node t are removed or merged if $\frac{\Delta D(S_t)}{\Delta R(S_t)} < \lambda_{mv}$. Consider a node whose immediate children are leaf nodes *after* the MV (quad) tree is pruned (refer to Fig. 5.1). For simplicity, let us denote the distortion associated with this node by D , and those with its children nodes by D_i , $i = 1, 2, 3, 4$. Also assume that the bitrate for MV tree mapping is negligible and that the bitrate for each MV is constant, denoted by r . For such a node surviving tree pruning, we have

$$\frac{D - \sum_{i=1,2,3,4} D_i}{4r - r} = \frac{D - \sum_{i=1,2,3,4} D_i}{3r} \geq \lambda_{mv} \tag{5.5}$$

or equivalently,

$$D \geq \sum_{i=1,2,3,4} D_i + 3r\lambda_{mv} \tag{5.6}$$

Assuming MSE is used as the distortion measure, then

$$N\sigma^2 \geq \sum_{i=1,2,3,4} N_i\sigma_i^2 + 3r\lambda_{mv} \tag{5.7}$$

where N and σ^2 are the number of pixels and the MSE of the parent motion block, respectively, and N_i and σ_i^2 are the number of pixels and the MSE of the respective children block, respectively. Obviously, $N = \sum_{i=1,2,3,4} N_i$ and N is proportional to the motion block size.

A few interesting properties can be observed from (5.7):

- If the block size is large, a small difference between the average MSE of the children blocks and the MSE of the parent block will ensure the survival of children blocks, which means smooth regions can use large blocks as splitting does not help reduce MSE significantly.
- If the block size is small, the difference between the average MSE of the children blocks and the MSE of the parent block needs to be large to ensure that children blocks are not pruned, which means smooth regions are less likely.
- If block size is too small, even for busy regions, which often require very small motion blocks to reduce MC residues, children nodes may have to be pruned.

Indeed, we observe that homogeneous regions are often associated with large motion blocks, while regions with lots of motion associated with small motion blocks. For illustration, Fig. 5.4(a) shows a frame taken from the *Bus* sequence, with final motion blocks indicated with white grids. The temporal high-pass frame, i.e., MC residue, is shown in (b). By superimposing (a) and (b), we are able to visually associate the residues with the motion blocks, as shown in (c). It indeed shows a general trend that when a large motion block is used, the residue is often small; on the other hand, when a small motion block is used, very often significant residue results.

5.2 Set-partitioning According to Motion Blocks

Morphological coders (see Section A.6) essentially try to capture the clustering of significant wavelet coefficients around textured regions using morphological dilation [57][66], so that the *significant* set will be dominated by ones (or non-zero values) and the *insignificant* set dominated by zeros. As a result, these two sets are each modeled with a more skewed probability distribution than the original distribution, and the total entropy of the two sets should be lower than that of the original set [40][67]. Therefore, a coding gain can be expected by this improved probability modeling.

As the temporal high-pass frames exhibit different characteristics (magnitude, homogeneity, etc.) in motion blocks of different sizes, we hypothesize that a reduction in entropy may be achieved by partitioning the wavelet coefficients into sets according to motion blocks, and if so, a coding gain may then be expected. Following this motivation, we can partition the wavelet coefficients into groups according to the size of the motion block to which they correspond. Of course, the exact procedure of partitioning is dependent on the specific coder. However, note that the proposed idea is only coupled to the ME algorithm, not to the decomposition structure, be it $t+2D$ or $2D+t$.

Let us establish some notations to facilitate the discussions below. Let K and L be positive integers, where L denotes the levels of spatial wavelet decomposition and K the total number of spatial subbands. Let k denote the spatial subband, $k \in \{0, 1, \dots, K\}$, with subband 0 being DC, $\{1, 4, \dots, 3L - 2\}$ for those in HL orientation, $\{2, 5, \dots, 3L - 1\}$ for those in LH orientation, and $\{3, 6, \dots, 3L\}$ for those in HH orientation, all with increasing frequency. Clearly, $K = 3L + 1$. An example for $L = 3$ is shown in Fig. 5.5(a). The frame indices in a group of pictures (GOP) comprising 16 frames after being subjected to four levels of MCTF are shown in Fig. 5.5(b). Furthermore, let j denote the set of all motion blocks of the same size, $j \in \{0, 1, \dots, M\}$, where set 0 corresponds to the largest motion blocks and set M to the smallest motion blocks. Also let S_k^{min} be the index to the

Figure 5.4: Illustration of motion estimation block size and residues. Produced using the HVSMBE algorithm of the MC-EZBC coder [17].

set of largest motion blocks for modelling in subband k , and S_k^{max} the index to the set of smallest motion blocks for modelling in subband k . Clearly, $S_k^{min} \geq 0$ and $S_k^{max} \leq M$.

This set-partitioning idea is apparently applied only to temporal high-pass frames, as it requires a direct mapping between the positions of wavelet coefficients and motion blocks (refer to Section 3.2). The partitioning may be customized for different temporal levels. When motion estimation is conducted with a frame closer to the reference frame in time, it is more likely to find homogeneous regions; on the other hand, when conducted with a frame further away in time (e.g., at lower levels of the temporal decomposition), the content may have changed so much that there are few homogeneous regions and hence many motion blocks of small sizes would result. Thus, S_k^{min} and S_k^{max} may be set to different values to avoid some sets having too few coefficients. For example, assume the motion blocks have 5 sizes (hence 5 sets). For the highest temporal level, we may choose $S_k^{min} = 0$ and $S_k^{max} = 3$ so that the first three sets cover the three sets of larger motion blocks in

Figure 5.5: Naming conventions for spatial and temporal subbands

a one-to-one correspondence, and the last set covers all motion blocks from sets 3 and 4. On the other hand, for the lowest temporal level, it might be more appropriate to choose $S_k^{min} = 2$ and $S_k^{max} = 4$, so that motion blocks in sets 0, 1, and 2 are grouped into one set, as the number of wavelet coefficients belonging to set 0 and/or 1 is very likely to be small.

5.3 Experiments

Experiments are conducted on MC-EZBC to confirm our hypothesis, as MC-EZBC precisely uses a hierarchical motion estimation method called *hierarchical variable size block matching* (HVSBM) [15]. The settings for MC-EZBC are listed in Table 5.1. With the listed and default settings, one GOP consists of 16 pictures, and 5-level spatial wavelet decomposition is used. The first experiment measures the entropy of the Y component of temporal high-pass frames with versus without set-partitioning aligned with motion blocks. The second experiment modifies the frame (image) coding module (i.e., EZBC [10]) of MC-EZBC so as to verify whether set-partitioning aligned with motion blocks can improve coding efficiency.

5.3.1 Experiment 1

As MC-EZBC uses 64×64 top-level motion blocks, S_k^{min} and S_k^{max} are set to 1 and 3, respectively, so that 64×64 and 32×32 blocks are grouped into set 1, 16×16 blocks into set 2, and 8×8

Table 5.1: Parameter settings for MC-EZBC

Parameter	Value
-format	YUV
-start	1
-last	288
-size	352 288 176 144
-framerate	30
-intra	NO
-denoise	NO
-motion	hvsbm
-search	1
-accuracy	quarter
-OPEN_GOP	YES
-tPyrLev	4

blocks and below into set 3. For simplicity, the same setting is used for all temporal levels. Since MC-EZBC employs bitplane coding of wavelet coefficients, one coefficient will no longer be tested for significance once its MSB bitplane has been tested. In other words, once the coefficient has tested significant its subsequent bitplanes will simply be transmitted without testing (refinement only). As the refinement bits are less context dependent than the MSB, we only measure the entropy of the significance information in individual bitplanes. Specifically, in order to accurately measure the entropy of significance information of coefficients at bitplane n , the already significant ones, whose magnitudes are no less than 2^{n+1} , should be excluded from consideration. The percentage of processed (i.e., included) over total coefficients in each subband is gathered during experiment to reflect how ‘coarse’ the equivalent quantization is. The proportion of processed coefficients partitioned into each set is also gathered. To compute the entropy of a source with histogram p , we use the following formula [40]:

$$H(p) = - \sum_n p(n) \log_2 p(n) \quad (5.8)$$

The steps of the first experiment performed *on each temporal high-pass frame* is summarized as follows:

1. Test bit n of each coefficient in each subband, and keep only those with magnitude strictly less than 2^{n+1} for further processing and discard the rest. Count the total number of kept coefficients in each subband.
2. For each subband, classify the coefficients surviving step 1, and count the number of coefficients assigned to each set.
3. For each subband, compute the percentage of processed coefficients against the total number, and the proportion of coefficients falling into each set, denoted as γ_1 , γ_2 , and γ_3 , respectively, against total number of *processed* coefficients.

4. For each subband, compute the following quantities:

- Original entropy estimate $H(p^{orig})$: this is done on the *processed* coefficients, by computing a normalized histogram p^{orig} and substituting into (5.8).
- Entropy estimate of each set $H(p^{s1})$, $H(p^{s2})$ and $H(p^{s3})$: this is done on coefficients in each partitioned set, similarly by computing normalized histograms p^{s1} , p^{s2} and p^{s3} , respectively, and substituting into (5.8).
- Composite entropy estimate $H(p^{comp})$: this is a weighted sum of entropy estimates of partitioned sets, i.e., $H(p^{comp}) = \sum_{i=1,2,3} \gamma_i H(p^{s_i})$, where γ_i is the obtained in step 3.
- Gain (reduction in entropy) expressed in percentage.

■

We ran the above steps on each bitplane separately. Apparently, as bitplane n decreases, more pixels will have been significant and hence be excluded from the test set. Since our goal is to show if there is any reduction in the entropy of significance information, we only report the results for bitplane 5, corresponding to a quantizer size of 32, and consistent results are obtained for other bitplanes as well. The detailed breakdown of experimental results for GOP 1 (frames 1-16) is tabulated in Appendix B for *Foreman* and in Appendix C for *Mobile*, arranged in a subband-centric manner. Each table shows the percentage of processed coefficients, the proportion and entropy of processed coefficients in each set, the average entropy, entropy reduction for each temporal high-pass frame, and also the entropy reduction averaged over the GOP.

The change in entropy for each subband averaged over frames by the proposed set-partitioning method for *Foreman* and *Mobile* is shown in Fig. 5.6, both for GOP 1 alone and for all 18 GOPs. As can be seen, there is indeed a reduction in entropy for the temporal high-pass frames using the proposed set-partitioning method. A general trend is also observed that the reduction becomes lower with increasing spatial frequency. Since the spatial subbands are of different sizes, we need to normalize the subband reductions to obtain the frame-wise reduction. Let G_k be the reduction in subband k , and A_k be the size of subband k . Then the normalized reduction G_N is calculated as:

$$G_N = \frac{\sum_{k=0,1,\dots,K} G_k * A_k}{\sum_{k=0,1,\dots,K} A_k} \quad (5.9)$$

With (5.9), the normalized reductions for *Foreman* and *Mobile* are 6.8% and 8.1%, respectively. However, whether this reduction translates into an improvement in coding efficiency is the question that we are ultimately interested in answering, while will be covered in next section.

5.3.2 Experiment 2

In MC-EZBC, each frame after MCTF is coded by image coder EZBC separately. We briefly introduce EZBC so as to give reader an idea of how it was modified for our experiment. In EZBC, a quadtree is built for each subband with the coefficients being the tree leaves and a parent node being the maximum magnitude of its four children. As a bitplane coder, EZBC progressively encodes

(a) GOP 1 of *Foreman* (b) GOPs 1-18 of *Foreman* (c) GOP 1 of *Mobile* (d) GOPs 1-18 of *Mobile*

Figure 5.6: Average reduction in entropy (of significance information) with set-partitioning. Each subband is averaged over all temporal high-pass frames in the respective GOP.

subband coefficients from the most significant bitplane toward the least significant bitplane. Each bitplane pass performs two basic operations: test all the insignificant quadtree nodes against the current bitplane threshold, and refine the pixels (quadtree leaves) already tested significant from the previous passes. Once a node is tested significant, it is split into four descendent nodes. Such traversing of quadtree, including significance testing and node splitting, proceeds recursively in top-down order until the bottom (pixel) level (refer to Section A.5 and Fig. A.12). Whenever a pixel tests significant, its sign is coded immediately. Significance test, sign and refinement bits are all coded by context-dependent arithmetic coding. A list of significant pixels (LSP) is defined for each subband, containing the list of pixels tested significant so far. One separate list of insignificant nodes (LIN) is defined for each quadtree level (including the pixel level) of each subband, containing the list of nodes from that level that are tested insignificant. Note that the LIN for the pixel level of a subband in fact contains the list of insignificant pixels. Both the LSP and LIN's are maintained separately for each subband. Readers are referred to [10] for a detailed description and the source code is available from the MPEG CVS server (as of 20 Dec. 2005).

We modify the context modelling for significance testing of both quadtree nodes and insignificant coefficients in the LIN's by separating the modelling tables (histograms) for each set partitioned

Table 5.2: PSNR (dB) comparison with vs. without set-partitioning according to motion blocks. LIP context models modified.

Rate(kbps)	300	500	700	1000	1300	1600	2000
Foreman							
Orig.	32.84	35.11	36.47	38.08	39.24	40.25	41.38
New	32.84	35.11	36.46	38.08	39.24	40.25	41.37
<i>diff.</i>	0.0	0.0	0.0	0.0	0.0	0.0	-0.01
Mobile							
Orig.	25.54	28.68	30.40	31.99	33.15	34.12	35.32
New	25.54	28.67	30.40	31.98	33.15	34.12	35.31
<i>diff.</i>	0.0	-0.01	0.01	-0.01	0.0	0.0	-0.01

Table 5.3: PSNR (dB) comparison with vs. without set-partitioning according to motion blocks. LIN context models modified.

Rate(kbps)	300	500	700	1000	1300	1600	2000
Foreman							
Orig.	32.84	35.11	36.47	38.08	39.24	40.25	41.38
New	32.84	35.11	36.47	38.08	39.24	40.25	41.37
<i>diff.</i>	0.0	0.0	0.0	0.0	0.0	0.0	-0.01
Mobile							
Orig.	25.54	28.68	30.40	31.99	33.15	34.12	35.32
New	25.54	28.68	30.40	31.99	33.15	34.12	35.32
<i>diff.</i>	0.0	0.0	0.0	0.0	0.0	0.0	0.0

according to the motion block they fall in. S_k^{min} and S_k^{max} are set to 1 and 3, respectively, as in experiment 1. It is straightforward to find the corresponding motion block of a coefficient according to the subband orientation and resolution. For a quadtree node, the corresponding motion block can be found by first finding its position in the subband domain, and then translating to the spatial domain. It is worthwhile to emphasize that this modification requires *no* additional information be transmitted to the decoder, because both encoder and decoder can infer all necessary information directly from the motion information that is already available before entropy encoding/decoding.

The experiment results for the Y component with this set-partitioning are summarized in Table 5.2 and Table 5.3, respectively. Unfortunately, the result is rather disappointing: it shows a zero or negative effect on PSNR performance. We discuss the reasons for this phenomenon in the next subsection.

5.3.3 Discussion

Our experiments show that with set-partitioning aligned with motion blocks, the entropy of significance information of subband coefficients is reduced for almost all bitplanes, but coding performance does not improve. This phenomenon is closely related to both the quadtree structure and

context modelling of EZBC.

Firstly, as bitplane coding starts with MSB and progresses towards LSB, the quadtree structure effectively prevents descendent nodes from being tested if their parent is insignificant, especially for high bitplanes. Therefore, only a small portion of nodes/coefficients are actually tested at high bitplanes, and the proportion should be much smaller than that indicated in experiment 1 (which does not take into account the quadtree structure). This reflects the efficiency of the quadtree structure in capturing local homogeneity. Of course, as bitplane coding progresses towards lower bitplanes, the effective quantizer size becomes smaller and there will be more nodes splitting and hence more significance testing.

Secondly, since the set-partitioning aligned with motion blocks is applied on top of the existing context modeling in EZBC, each significance test context is further divided into a few ones. As a result, the number of nodes/coefficients in each context may become quite small and hence insufficient to establish reliable statistics. The minimum expected code length will not be achieved until accurate source statistics has been accumulated. Therefore, poor coding efficiency may result due to too many contexts. This is known as a *context dilution* problem [68]. This problem may be overcome or at least mitigated by performing set-partitioning selectively; for example, it might be better to carry out set-partitioning only for lower bitplanes when there are more significance testings. It remains as a future work to investigate such possibilities.

Though experiment with MC-EZBC does not show PSNR improvement, there is no reason that we conclude set-partitioning will not work at all, or will not work with other tools. In fact, it is conjectured that the proposed idea of set-partitioning aligned with motion blocks may be more suitable for single bitrate coders based on simple scalar quantization. It may be combined with arithmetic coding without sophisticated data structures, modelling contexts and coding passes, in which case we obtain a simple and possibly also efficient embedded coder. This remains as a future research topic.

Chapter 6

Concluding Remarks and Future Work

In this thesis, we introduced the emerging *Motion-compensated wavelet coding* (MCWC) technology that achieves temporal, spatial and SNR scalability through a uniform framework, thanks to the inherent multi-resolution property of wavelets. The efficiency of MCWC coders lies in *motion-compensated temporal filtering* (MCTF) and embedded 3-D coding techniques. We close the thesis with a summary of the contributions of this research work and directions for future research.

6.1 Contributions of the Thesis

We studied the spatial scalability problem in the t+2D scheme, i.e., the degraded R-D performance for lower spatial resolutions, and proposed one decoder-side solution, one encoder-side solution and a third *practical* solution to address this problem (see Chapter 4). We have seen that, due to motion compensation, the temporal dimension and spatial dimension are coupled, instead of being independent. In particular, for the t+2D scheme, the R-D performance for lower spatial resolutions is adversely affected if inverse MCTF (IMCTF) is applied at the target lower spatial scale at the decoder. When reconstructing a lower spatial resolution video, the proposed decoder-side solution up-scales the spatially recovered frames to the original (full) resolution, applies IMCTF on the full resolution frames, and subsequently down-scales the reconstructed frames to the target lower resolution. Note that the decoder-side solution only involves modifying the existing decoder. The R-D performance for lower spatial resolutions can be greatly improved (by more than 1 dB for half resolution) at the cost of increased computational complexity.

The proposed decoder-side solution only mitigates the spatial scalability problem and cannot cancel the aliasing incurred during MCTF at the encoder. One encoder-side solution was then proposed after studying the fundamentals of MCTF. This solution works by making each spatial resolution self-contained in the sense that subbands in each spatial scale only depend on those in the lower scale or the same scale; hence the name *low-to-high lifting MCTF* (LTH-MCTF). Throughout the discussions that lead to LTH-MCTF, we highlighted that the fundamental difference between

the $t+2D$ scheme and the $2D+t$ scheme lies in which spatial bands are used to predict/update one spatial band in MCTF, rather than the apparent order of performing the spatial and temporal decompositions. We also established that the two seemingly dramatically different schemes can be realized by one another with proper control of the prediction/update signal.

Facing the fact that the best coding performance is not attainable simultaneously for all spatial resolutions with a fully embedded bitstream, and optimizing performance for one spatial resolution leads to reduced performance for the rest, we proposed a *practical* solution, named *pyramidal $t+2D$* , that does not produce a fully embedded bitstream and that instead applies $t+2D$ coding to each level of the frame pyramids to produce one separate bitstream with optimal performance for each spatial resolution. Though the coded bitstream is overcomplete, this practical solution can be justified by the practical constraints in an end-to-end video system: transmission costs much more than storage.

We observed that the temporal high-pass frames in general do not resemble natural images, and applying common embedded image coding methods originally designed for natural images may not be optimal in terms of complexity and performance. We studied the statistical properties of the temporal high-pass frames when variable size motion estimation/compensation is employed, and discovered that the motion-compensated prediction residues exhibit different statistics in motion blocks of different sizes (see Chapter 5). We hence propose to partition the wavelet coefficients into groups according to the size of the motion block to which they correspond, for better contextual modelling. It is observed that the entropy of significance information (when some bitplane is tested) is reduced. However, in our experiments with MC-EZBC, the final coding performance does not improve despite the reduction in entropy. Explanations for this phenomenon were offered.

6.2 Future Research Directions

We have established the fundamental difference between the $t+2D$ scheme and the $2D+t$ scheme in terms of the source of the prediction/update signal. However, their R-D performance is unlikely to be identical, due to other factors in the coding system. Given that the same tools (e.g., wavelet filters, texture coding, entropy coding) are used, it remains to see which realization performs better. This may help design practical coding systems when both the R-D performance and the computational complexity have to be jointly considered.

The study on set-partitioning the wavelet coefficients is preliminary, and better partitioning methods may be devised following the idea of aligning with motion blocks. This idea is in fact very general; that is, it is in no way limited to MCWC coders, but potentially applicable to any video coder using variable size block motion estimation/compensation and entropy coding, especially single bitrate coders based on scalar quantization. Thus, it remains interesting to see how set-partitioning aligned with motion blocks can be applied to other types of coders.

Appendix A

Review of Embedded Wavelet Image Coding

Embedded wavelet image coding (EWIC) has demonstrated superior compression performance over traditional DCT-based coders. For example, early embedded wavelet coders like EZW [8] and SPIHT [9] already can outperform JPEG [69, 70]. More recent development in EWIC has produced coders with even higher performance, e.g., EBCOT [13], EZBC [10]. One noteworthy feature of EWIC is that the superior compression performance is obtained with an embedded bitstream that is *both SNR and resolution scalable*. In this chapter, we will first introduce the fundamental concepts of *embedded coding* and *entropy rate* that constitute the theoretical basis of EWIC in Section A.1 and Section A.2, respectively. The basic tools employed in EWIC, namely *wavelet transform*, *bit-plane coding* and *set-partitioning*, are introduced in Section A.3. Set-partitioning in EWIC is to organize the wavelet coefficients into some structures to efficiently exploit their correlations and facilitate context modelling for entropy coding. In fact, the major distinction between EWIC coders lies in their strategies of forming such structures. We broadly classify popular EWIC coders into three categories, namely *zerotree*, *zero-block* and *morphological*, which are reviewed in Section A.4, Section A.5 and Section A.6, respectively.

A.1 The Embedding Principle

Embedded coding may be defined as follows: if two bitstreams, A and B , produced by the encoder, have size M and N bits each, with $M > N$, then bitstream B is identical to the first N bits in bitstream A .

The above definition implies that the bitstream of a larger size always produces a better approximation to the original signal, regardless of the distortion measure used. In the context of data compression, a *fully embedded bitstream* should be capable of reconstructing the signal with the best quality for every bitrate simultaneously, such that a truncated bitstream is also the best for its bitrate. Consequently, it implies that bits that can reduce more distortion should be embedded into the stream *before* the rest. This is commonly called the *embedding principle* [71].

Figure A.1: Illustration of the embedding principle. The dashed curve ($c \rightarrow b \rightarrow a$) represents a better R-D performance than the solid curve ($a \rightarrow b \rightarrow c$).

Let ΔD_i be the reduction in distortion by sending ΔR_i bits for data i . By embedding principle, the data should be sent in decreasing order of $\frac{\Delta D_i}{\Delta R_i}$. For instance, consider the two curves shown in Fig. A.1, which represent two orders of sending the same data sets. Apparently, if all data a , b , and c are received, the bitstream always reaches the *rate-distortion* (R-D) point (R', D') . However, if the bitstream is truncated before rate R' , i.e., some data has to be dropped, the ordering indicated by the dashed line will reduce more distortion, hence a preferred choice in embedded coding.

A.2 Entropy Rate

In information theory, *entropy* is a measure of uncertainties associated with a random variable [40]. It quantifies the information contained in a message (represented as a random variable or a series of random variables) and is the minimum message length (in terms of bits) necessary to communicate the information. Formally, the entropy of a discrete random variable X with alphabet $\mathbb{A}$ and probability mass function (PMF) f_X is defined as

$$H(X) = - \sum_{x \in \mathbb{A}} f_X(x) \log_2 f_X(x) \quad (\text{A.1})$$

The entropy rate of a discrete random process X_i with joint PMF $f_X(x_1, x_2, \dots, x_n)$, $(x_1, x_2, \dots, x_n) \in \mathbb{A}^n$, is defined as

$$\begin{aligned} H(\mathbb{A}) &= \lim_{n \rightarrow \infty} \frac{1}{n} H(X_1, X_2, \dots, X_n) \\ &= - \frac{1}{n} \sum \dots \sum_{(x_1, x_2, \dots, x_n) \in \mathbb{A}^n} f_X(x_1, x_2, \dots, x_n) \log_2 f_X(x_1, x_2, \dots, x_n) \end{aligned} \quad (\text{A.2})$$

when the limit exists.

The theoretical bound $H(\mathbb{A})$ can be approached by encoding coefficients in blocks of increasing size, known as m -th order extension of the source alphabet. However, the alphabet size $|\mathbb{A}|^m$ grows

exponentially. For example, for groups of 4×4 coefficients with 8-bit depth, the alphabet size is 256^{16} .

A related quantity for entropy rate of a random process, known as the conditional entropy of the last random variable given the past, is defined as

$$\begin{aligned} H_c(\mathbb{A}) &= \lim_{n \rightarrow \infty} \frac{1}{n} H(X_n | X_1, X_2, \dots, X_{n-1}) \\ &= - \sum \dots \sum_{(x_1, x_2, \dots, x_n) \in \mathbb{A}^n} f_X(x_n | x_1, x_2, \dots, x_{n-1}) \log_2 f_X(x_n | x_1, x_2, \dots, x_{n-1}) \quad (\text{A.3}) \end{aligned}$$

when the limit exists.

The two limits in (A.2) and (A.3) exist and are equal for a stationary random process [40]. The latter definition offers an alternative way of approaching the theoretical bound by conditional entropy coding of increasing order. This fact has been widely exploited by context modelling in many coding systems. In particular, for embedded wavelet image coders, efficient structures have been found to take advantage of correlations among wavelet coefficients to achieve good coding efficiency.

A.3 Overview of Embedded Wavelet Image Coding

The block diagram of an embedded wavelet image coding system is depicted in Fig. A.2. At the encoder, the image signal is first transformed into the wavelet domain by two-dimensional discrete wavelet transform (2D-DWT). Specific data structures are then formed on the wavelet coefficients to efficiently exploit the correlations among them and facilitate the context modelling in the following bitplane/entropy coding stage. The data structures often prioritize the wavelet coefficients according to their *importance* (based on some measure, e.g., distortion reduction per coded bit), so that the more important coefficients will be coded and transmitted before the less important ones. This is just the basic idea and probably over-simplified, as in EWIC instead of treating each wavelet coefficient altogether, the coefficients are successively quantized with decreasing quantization step sizes, and one coding pass is carried out for each quantization level. Through this way, a progressive approximation to the original image is built up. The decoder reconstructs the received bitstream by precisely reversing the coding process the encoder has taken. The difference is that the decoder *learns* new information bit by bit from the received bitstream and updates its state accordingly, while the encoder learned the same information directly from the input signal. Hence, the decoder would be at the same state as the encoder was, after consuming precisely those bits outputted by the encoder.

Pioneered by the *embedded zerotree wavelet* (EZW) coder by Shapiro [8], many image coders along the line of embedded wavelet coding have been devised [9, 72, 57, 13, 10, 66]. Sharing a similar system structure as that shown in Fig. A.2, the major distinction among different embedded wavelet image coders lies in their strategy of organizing the wavelet coefficients into efficient data structures. Hence, after introducing the important tools used in an EWIC system in the remaining of this section, subsequent sections will concentrate on representative EWIC coders in the literature.

Figure A.2: Embedded wavelet image coding system

Figure A.3: Wavelet filter spectrum

A.3.1 Wavelet Transform

In image compression, wavelet bases (or filters in the perspective of signal processing) are normally chosen to have a short support, in order to be computationally efficient as well as to localize texture features. Fig. A.3 illustrates the spectrum of an ideal 2-D wavelet basis, where each dimension is evenly divided into a low-pass band and a high-pass band. To maximally remove redundancies among the wavelet coefficients, several levels of decomposition are performed with maximal downsampling. The most commonly used decomposition structure in image coding is the multi-level (critically sampled) dyadic decomposition, where the 2D transform is first applied to the input image, and is then iteratively applied to the LL band of the previous decomposition. This process is illustrated in Fig. A.4 in the case of a *separable* 2D-DWT, where the transform takes place first in horizontal direction and then in the vertical direction. Fig. A.5 shows the 10 subbands obtained after a three-level dyadic decomposition. Note that frequency resolution increases as the frequency decreases.

It is worthwhile to highlight a few important properties of wavelet transform in the context of

Figure A.4: Separable wavelet decomposition. Input image is initially connected to the processing banks and is disconnected after the first level of decomposition. LL band is then iteratively connected as input.

Figure A.5: Subbands after three-level dyadic decomposition

image coding:

- The iterative nature of dyadic wavelet decomposition inherently offers a multi-resolution representation of the original image, which makes spatial scalability a simple matter of gathering the wavelet coefficients from respective subbands. For example, the subbands $\{LL_3, HL_3, LH_3, HH_3\}$ correspond to an image of $1/4$ original size, and subbands $\{LL_3, HL_3, LH_3, HH_3, HL_2, LH_2, HH_2\}$ to an image of $1/2$ original size.
- At one scale, the wavelet basis is a set of translated versions of the same function that has a short support [21]. This leads to the wavelet transform offering space resolution on top of frequency resolution. Hence wavelet transform is able to localize texture features of an image, a distinct characteristic from other transforms with infinitely long basis functions like *discrete cosine transform* (DCT) [73, 74, 75, 76]. For illustration, the wavelet coefficients of the *Lena* image (shown in Fig. A.6) transformed using the biorthogonal 9/7 wavelets are shown in Fig. A.7, with subbands organized in the layout as in Fig. A.5. It is evident that the shapes of the hat and hair are visible at different scales.

A.3.2 Bitplane Coding

A wavelet coefficient c can be represented in *sign-magnitude* form of $c = \text{sgn}(c)|c|$, where $\text{sgn}(\cdot)$ returns the sign bit, and the magnitude $|c|$ can be expressed in binary representation as $|c| = \sum_n b_n 2^n$, $n \in \{0, 1, \dots, N\}$, with N being the left-most bit and $b_n \in \{0, 1\}$. The basic idea of bitplane coding is to transmit the binary representation of the wavelet coefficient's magnitude from the *most significant bit* (MSB) to the *least significant bit* (LSB), so that the original image is progressively approximated by the transmitted bitstream. The sign bit is transmitted separately before transmitting the magnitude. Transmitting up to bitplane n is equivalent to quantizing the coefficients by a quantizer of step size 2^n with a central dead zone of twice the step size; that is,

$$Q_n(x) = \begin{cases} 0 & \text{if } -2^n < x < 2^n \\ q & \text{if } q2^n \leq x < q2^{n+1} \\ -q & \text{if } -q2^{n+1} < x \leq -q2^n \end{cases} \quad (\text{A.4})$$

where q is a positive integer. It is easy to see that transmitting one more bitplane is equivalent to reducing the quantization step by half. The major components of bitplane coding are illustrated in Fig. A.8.

Scanning the coefficients in a given bitplane in a pre-defined order does not necessarily produce the best rate-distortion (R-D) result. Thus, in order to make the final bitstream achieve the best quality possible at any bitrate, EWIC coders divide each bitplane coding pass into several fine sub-passes to optimize R-D performance. As illustrated in Fig. A.9, sub-passes for each bitplane will result in a rate-distortion curve that is closer to the optimal one. While this is very much coder-dependent, the rule-of-thumb is to give priority to bits that will more likely reduce distortion.

In image coding, it is common that a set of coefficients are tested for significance with respect to a threshold. We say that a set $\mathcal{G}$ is *significant* with respect to T if

Figure A.6: The original Lena image (512x512)

Figure A.7: Wavelet coefficients of the Lena image (512x512). Three levels of dyadic decomposition with biorthogonal 9/7 wavelets are applied, and coefficients are rescaled for display.

Figure A.8: Bitplane coding illustration

Figure A.9: Rate-distortion curve of bitplane coding. A 'o' represents an attainable R-D point with a bitplane full-pass, while a '★' represents an R-D point that is only attainable by a fine sub-pass within a bitplane.

$$\max_{(i,j) \in \mathcal{G}} \{|c_{i,j}|\} \geq T, \quad (\text{A.5})$$

and *insignificant* otherwise. In bitplane coding, for bitplane n , we can write the significance test function as

$$S_n(\mathcal{G}) = \begin{cases} 1, & \text{if } 2^n \leq \max_{(i,j) \in \mathcal{G}} \{|c_{i,j}|\} < 2^{n+1} \\ 0, & \text{otherwise} \end{cases} \quad (\text{A.6})$$

That is, the test returns true when the coefficients first become significant. This significance test is widely used in embedded coders for signalling the coefficients in a particular structure (set) are insignificant, which will be described in the later sections.

A.3.3 Set-partitioning

As mentioned, the major distinction between EWIC coder lies in how the wavelet coefficients are organized into data structures to efficiently exploit their correlations and facilitate context modelling, which may be viewed as an approximation to high-order conditional entropy coding. An early class of embedded wavelet coders including EZW [8] and SPIHT [9] arrange wavelet coefficients across spatial scales into a tree structure known as *zerotrees*, with one coefficient having four descendants located at the corresponding area in the next finer scale. The zerotree can efficiently capture the inter-scale dependencies among the wavelet coefficients in that when the parent coefficient is below the threshold, it is often true that all its descendants are also below the threshold. This relationship is often true because most images have a decaying energy spectrum. Another class of coders arrange wavelet coefficients into a quadtree for each subband, with each node assigned the maximum magnitude of its offspring. Such quadrees are efficient in capturing the local features. For example, if a node is below the threshold, then it immediately follows that all coefficients covered by the node are below the threshold. These coders are commonly referred to as *zero-block* coders, e.g., SPECK [72], EZBC [10]. Both *zerotree* and *zero-block* based coders can be broadly classified as *zero-set* coders, as they all try to represent a set of coefficients with a single bit. An interesting class of coders known as morphological coders partition wavelet coefficients dynamically into *significant* and *insignificant* sets for more efficient statistical modeling and hence entropy coding [57, 66]. All three classes are broadly termed as *set partitioning* based coders in this thesis, though they have different mechanisms.

Note the difference between set-partitioning of wavelet coefficients and block segmentation in the sample domain. The former groups coefficients with similar (or highly likely similar) properties, and the latter forms units for transformation. Hence set-partitioning wavelet coders do not exhibit blocking artifacts commonly arising from block-based transform coding.

A.4 Zerotree Coding

As mentioned in Section A.3.1, due to the short support of wavelet filters, spatial features are localized in the wavelet domain. In a dyadic decomposition, it is therefore reasonable to expect

Figure A.10: Parent-children relationship in a spatial orientation tree

some similarities across scales, as demonstrated in Fig. A.7. This leads to the idea of grouping wavelet coefficients in different scales corresponding to the same location into a tree structure. Fig. A.10 shows the parent-children relationship in such a spatial orientation tree. EZW [8] and SPIHT [9] utilize such tree structures to efficiently remove the inter-scale correlations. Specifically, if a coefficient is quantized to zero and so are all its descendants in the finer scales, a single bit is transmitted to indicate that a *zerotree* is present, so that all the descendants are automatically represented. The existence of zerotrees in the wavelet domain can be reasoned by considering the fact that natural images are dominated by low frequencies, i.e., their spectra decay as frequency increases, as exemplified in Fig. A.11. Hence, for the same spatial location, if coefficients in the lower frequency subband are below the threshold (quantizer size), there is a good chance that the corresponding coefficients in the finer scales will also be below the threshold.

To produce an embedded bitstream, bitplane coding is often employed in zerotree coders. We know from Section A.3.2 that quantization is effectively performed on the wavelet coefficients for each bitplane. With a decaying spectrum, it is apparent that the higher the bitplane, the more likely the existence of zerotrees. As progressing from high bitplanes to low bitplanes, the depth of zerotrees starts to decrease and some zerotrees start to vanish.

It is probably worthwhile to mention a non-embedded zerotree wavelet image coder proposed by Xiong et al. known as *space-frequency quantization* (SFQ) [77], so that the idea of zerotree can be understood from an alternative perspective. The basic idea of SFQ is to quantize a subset of wavelet coefficients to zero (known as *spatial quantization*), and quantize surviving coefficients by a standard scalar quantizer. Spatial quantization in SFQ is performed on spatial orientation trees of wavelet coefficients, hence it is referred to as *zerotree quantization*. Applying scalar and zerotree quantization modes in a jointly optimal fashion, SFQ achieves substantial coding gain over the

Figure A.11: Illustration of the decaying spectra of images. (a) and (c) are both 64x64 blocks from the Lena 512x512 image, with (a) having few spatial details and (c) having many spatial details. (b) is the magnitude of the DCT coefficients of (a), and (d) is that of (c).

basic zerotree scheme of EZW in terms of PSNR. Despite the improvement in PSNR, its subjective quality is not clearly superior. This is due to the fact that the objective distortion measure used in SFQ, *mean squared error* (MSE), is not perceptually accurate. While bits invested into the high frequency regions can reduce MSE, the subjective quality of these high frequencies is much less important than that of low frequencies. Hence the gain in PSNR does not necessarily translate into a gain in subjective quality.

A.5 Zero-block Coding

Zero-block based coders differ from zerotree coders in that a tree structure is constructed for each subband. Quadtrees are often used because of their simplicity in concept and implementation. EZBC [10] and SPECK [72] are two coders utilizing quadtrees. In one subband, the coefficients are leaves of the quadtree, and nodes are recursively created as the maximum magnitude of their four children. It is clear that the root of the quadtree is equal to the maximum magnitude of the coefficients in this subband. All nodes in the quadtree are initially insignificant. When coding a bitplane, the quadtree is traversed top-down, transmitting a 0 if a node is tested insignificant and 1 if it is tested significant. The children of a significant node are recursively tested. A simple example illustrating the construction and traversing of a quadtree is shown in Fig. A.12. Such a quadtree structure is especially efficient in coding homogeneous regions with little fluctuation in magnitude,

Figure A.12: Quadtree construction and traversing. Magnitudes of coefficients are shown in (a). Bit 1 (i.e., MSB of all) of coefficients is shown in (b). Bitplane 1 is being transmitted, i.e., quantizer size is 2. Generated bits for each quadtree level in top-down order are 1, 0011, and 00010010, respectively.

as only a single bit is needed to signal that all the coefficients in the region covered by a node are insignificant.

A.6 Morphological Coding

In mathematics, *morphological dilation* is a way of growing a set [78]. let S be a set upon which the dilation operation is to be applied, let B be some structuring element, and let $\oplus$ denote the morphological dilation operation. Then the *dilated* set $B \oplus S$ is defined to be the union of all points falling under the support of the structuring element B , when B is centered at each point in S . Intuitively, dilation operation results in an enlarged set containing the original S , plus a few nearby elements.

The basic idea of morphological coding is to first find a significant coefficient in a predefined scanning order. With this newly found significant coefficient, a step of morphological dilation is performed and neighboring coefficients are examined for significance. If new significant coefficients are found, the process is recursively applied on each of them; otherwise the process stops and scanning of coefficients resumes, until the whole subband is exhausted. In [57], Servetto et al. investigated the statistical properties of wavelet coefficients (except the DC band) of images and proposed two coding methods based on morphology. Their experiments with the Lena image (see A.6) showed that with a five-level wavelet decomposition, the entropy in subbands $\{LL_1, HL_1, LH_1, HH_1\}$ (refer to Fig. A.5 for notation) is reduced by about 13% on average through a simple intra-band partitioning of wavelet coefficients into two sets, *significant* and *insignificant*, with respect to a threshold. Inter-band dependencies are exploited at no cost by predicting a coefficient to be significant if its parent coefficient is significant, and to be insignificant otherwise, based on the spatial parent-children

Figure A.13: Illustration of dilation-run coding (from [66]). (h) is the structuring element used, (a) to (g) illustrate the run-length coding of positional information of the seeds and the dilation operation on each significant coefficient. Note that the MSB of the run-length of 0's, i.e., the binary number before '+' or '-' sign, is omitted without compromising decodability.

relationship (see Fig. A.10). Experiments showed that parent-children prediction reduces entropy by about 11%. It should be noted that the prediction in this work is different from that in zerotree based coders in that it directly exploits the clustering of significance across scales.

Wu and He proposed an embedded morphological image coder named Dilation-run [66] that combines bitplane coding with set-partitioning by morphological dilation. In Dilation-run, the positional information of the seed of a cluster, i.e., the original significant coefficient of the cluster, is run-length coded. Fig. A.13 illustrates how dilation-run run-length codes the positional information of the seeds (significant coefficients) and progressively clusters significant coefficients by dilation operation.

Such a partitioning scheme using morphological dilation works in practice due to the fact that significant coefficients are often clustered locally (e.g., around edges). More fundamentally, the idea of partitioning is to have two sets with distinct statistical properties which can lead to more efficient entropy coding. Specifically, the significant subset is expected to be dominated by non-zero values, while the insignificant subset to be dominated by zeros, so that skewed probability distributions would result. Entropy coding following this partitioning will be fed with a source with a probability distribution like that in Fig. A.14(b) instead of that in Fig. A.14(a). It is well known that entropy coders can more efficiently code a source with such skewed statistics [67].

A.7 Discussion

Throughout the above descriptions, it should be clear that *zero-set* and *morphological* coders are devised based on very different principles, although they both involve some *set-partitioning* mechanism. *Zero-set* coders attempt to represent a set of coefficients with a single bit, where the

Figure A.14: Probability distribution functions affect compressibility. A source with probability density function (PDF) in (a) is much harder to code than that with PDF in (b).

grouped coefficients are across scales in case of *zerotree* coders and are in a local region of the same subband in case of *zero-block* coders. On the other hand, *morphological* coders attempt to dynamically classify coefficients into significant and insignificant sets that each have a more skewed probability distribution with peak at a different value, so that the efficiency of entropy coding is improved.

Appendix B

Experiment Results: Entropy of
Foreman (GOP 1) with vs.
without Set-partitioning According
to Motion Blocks

[illegible][illegible]

[illegible][illegible]

Table B.9: Foreman Subband 8

[illegible]

Table B.10: Foreman Subband 9

[illegible]

[illegible][illegible]

Table B.13: Foreman Subband 12

[illegible]

Table B.14: Foreman Subband 13

[illegible]

Appendix C

Experiment Results: Entropy of Mobile (GOP 1) with vs. without Set-partitioning According to Motion Blocks

[illegible][illegible]

Table C.3: Mobile Subband 2

[illegible]

Table C.4: Mobile Subband 3

[illegible]

Table C.9: Mobile Subband 8

[illegible]

Table C.10: Mobile Subband 9

[illegible]

Table C.11: Mobile Subband 10

[illegible]

Table C.12: Mobile Subband 11

[illegible]

[illegible][illegible]

Bibliography

- [1] Coding of Audio-Visual Objects - Part 2: Visual. *ITU-T Recommendation H.262 and ISO/IEC 13818-2 (MPEG-2)*, 1994.
- [2] Generic Coding of Moving Pictures and Associated Audio Information - Part 2: Video. *ISO/IEC 14496-2(MPEG-4 Visual Version 1)*, Apr. 1999.
- [3] Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264/ISO/IEC 14496-10 AVC). *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, JVT-G050*, 2003.
- [4] J.-R. Ohm. Advances in Scalable Video Coding. *Proc. of the IEEE*, 93(1):42–56, Jan. 2005.
- [5] P. J. Burt and E. H. Adelson. The Laplacian Pyramid as a Compact Image Code. *IEEE Trans. on Communications*, 31(4):532–540, Apr. 1983.
- [6] J. Arnold, M. R. Fracter, and Y. Wang. Efficient Drift-Free Signal-to-Noise Ratio Scalability. *IEEE Trans. Circ. Syst. Video Tech.*, 10(1):70–82, Feb. 2000.
- [7] W. Li. Overview of Fine Granularity Scalability in MPEG-4 Video Standard. *IEEE Trans. Circ. Syst. Video Tech.*, 11(3):301–317, Mar. 2001.
- [8] J. M. Shapiro. Embedded Image Coding Using Zerotrees of Wavelet Coefficients. *IEEE Trans. Signal Processing*, 41(12):3445–3462, Dec. 1993.
- [9] A. Said and W. A. Pearlman. A New, Fast, and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees. *IEEE Trans. Circ. Syst. Video Tech.*, 6(3):243–250, Jun. 1996.
- [10] S.-T. Hsiang. Embedded Image Coding Using Zeroblocks of Subband/Wavelet Coefficients and Context Modelling. In *Proc. Data Compression Conf.*, pages 83–92, Mar. 2001.
- [11] Information Technology - JPEG2000 Image Coding System: Core Coding System. *ISO/IEC International Standard 15444-1, ITU Rec. T.800*, Aug. 2002.
- [12] M. Rabbani and R. Joshi. An Overview of the JPEG2000 Still Image Compression Standard. *Signal Processing:Image Communication*, 17:3–48, 2002.
- [13] D. Taubman. High Performance Scalable Image Compression with EBCOT. *IEEE Trans. Image Processing*, 9(7):1158–1170, Jul. 2000.

- [14] J.-R. Ohm. Three-dimensional Subband Coding with Motion Compensation. *IEEE Trans. Image Processing*, 3(5):559–571, Sep. 1994.
- [15] S.-J. Choi and J. W. Woods. Motion Compensated 3-D Subband Coding of Video. *IEEE Trans. Image Processing*, 8(2):155–167, Feb. 1999.
- [16] S.-T. Hsiang, J. W. Woods, and J.-R. Ohm. Invertible Temporal Subband/Wavelet Filter Banks with Half-pixel-accurate Motion Compensation. *IEEE Trans. Image Processing*, 13(8):1018–1028, Aug. 2004.
- [17] S.-T. Hsiang and J. W. Woods. Embedded Video Coding Using Invertible Motion Compensated 3-D Subband/Wavelet Filter Bank. *Signal Processing:Image Communication*, 16:705–724, 2001.
- [18] J. Xu, Z. Xiong, and Y.-Q. Zhang. Three-Dimensional Embedded Subband Coding with Optimized Truncation (3-D ESCOT). *Applied and Computational Harmonic Analysis*, 10:290–315, 2001.
- [19] J. C. Ye and M. van der Schaar. Fully Scalable 3D Overcomplete Wavelet Video Coding Using Adaptive Motion Compensated Temporal Filtering. In *Visual Communications and Image Processing*, pages 1169–1180, Jun. 2003.
- [20] S. Mallat. A Theory for Multiresolution Signal Decomposition: The Wavelet Representation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(7):674–693, Jul. 1989.
- [21] C. S. Burrus, R. A. Gopinath, and H. Guo. *Introduction to Wavelets and Wavelet Transforms: A Primer*. Prentice Hall, 1998.
- [22] P.P. Vaidyanathan. *Multirate Systems and Filter Banks*. Prentice-Hall, Englewood Cliffs, NJ, 1993.
- [23] G. Strang and T. Nguyen. *Wavelets and Filter Banks*. Wellesley-Cambridge, Cambridge, MA, 1996.
- [24] F. Mintzer. Filters for Distortion-Free Two-Band Multirate Filter Banks. *IEEE Trans. Acoust. Speech Signal Processing*, 33(6):626–630, Jun. 1985.
- [25] M.J.T. Smith and T.P. Barnwell III. Exact Reconstruction for Tree-structured Subband Coders. *IEEE Trans. Acoust. Speech Signal Processing*, 34(6):431–441, Jun. 1986.
- [26] M. M. Vetterli and J. Kovacevic. *Wavelets and Subband Coding*. Prentice-Hall, Englewood Cliffs, NJ, 1995.
- [27] S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, 2nd edition, 1999.
- [28] W. Sweldens. The Lifting Scheme: A Construction Of Second Generation Wavelets. *SIAM J. Math. Anal.*, 29(2):511–546, Mar. 1998.
- [29] W. Sweldens. Lifting Scheme: A New Philosophy in Biorthogonal Wavelet Constructions. *Proc. of SPIE*, 2569:68–79, 1995.

- [30] I. Daubechies and W. Sweldens. Factoring Wavelet Transforms into Lifting Steps. *J. Fourier Anal. Appl.*, 4(3):245–267, 1998.
- [31] J.-R. Ohm. Motion-compensated Subband Coding with Multiresolution Representation of Motion Parameters. In *Proc. IEEE Int. Conf. Image Processing*, pages 250–254, 1994.
- [32] J. W. Woods and G. Lilienfeld. A Resolution and Frame-rate Scalable Subband/Wavelet Video Coder. *IEEE Trans. Circ. Syst. Video Tech.*, 11(9):1035–1044, Sep. 2001.
- [33] A. Secker and D. Taubman. Lifting-based Invertible Motion Adaptive Transform (LI-MAT) Framework for Hhighly Sscalable Video Compression. *IEEE Trans. Image Processing*, 12(12):1530–1542, Dec. 2003.
- [34] A. Secker and D. Taubman. Highly Scalable Video Compression with Scalable Motion Coding. *IEEE Trans. Image Processing*, 13(8):1029–1041, Aug. 2004.
- [35] F. Verdicchio, Y. Andreopoulos, T. Clerckx, J. Barbarien, A. Munteanu, J. Cornelis, and P. Schelkens. Scalable Video Coding Based on Motion-Compensated Temporal Filtering: Complexity and Functionality Analysis. In *Proc. IEEE Int. Conf. Image Processing*, 2004.
- [36] D. S. Turaga, M. van der Schaar, and B. Pesquet-Popescu. Complexity Scalable Motion Compensated Wavelet Video Coding. *IEEE Trans. Circ. Syst. Video Tech.*, 15(8):982–993, Aug. 2005.
- [37] B.-J. Kim and W. A. Pearlman. An Embedded Wavelet Video Coder Using Three-dimensional Set Partitioning in Hierarchical Trees. In *Proc. Data Compression Conf.*, pages 251–260, Mar. 1997.
- [38] Draft Status Report on Wavelet Video Coding Exploration. *ISO/IEC JTC1/SC29/WG11, N7571*, Oct. 2005.
- [39] New Prediction Schemes for Scalable Wavelet Video Coding. *ISO/IEC JTC1/SC29/WG11, M12642*, Oct. 2005.
- [40] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, 1991.
- [41] G. J. Sullivan and R. L. Baker. Motion Compensation for Video Compression Using Control Grid Interpolation. In *Proc. IEEE Int. Conf. Acoust. Speech Signal Processing*, volume 4, pages 2713–2716, Apr. 1991.
- [42] Y. Nakaya and H. Harashima. Motion Compensation Based on Spatial Transformations. *IEEE Trans. Circ. Syst. Video Tech.*, 4(3):339–356,366–367, Jun. 1994.
- [43] C.-L. Huang and C.-Y. Hsu. A New Motion Compensation Method for Image Sequence Coding Using Hierarchical Grid Interpolation. *IEEE Trans. Circ. Syst. Video Tech.*, 4(1):42–52, Feb. 1994.

- [44] L. Luo, S. Li, Z. Zhuang, and Y.-Q. Zhang. Motion Compensated Lifting Wavelet and Its Application in Video Coding. In *Proc. IEEE Int. Conf. Multimedia Expo*, pages 365–368, Aug. 2001.
- [45] C.-Y. Chen, C.-T. Huang, Y.-H. Chen, S.-Y. Chien, and L.-G. Chen. System Analysis of VLSI Architecture for 5/3 and 1/3 Motion-Compensated Temporal Filtering. *IEEE Trans. Signal Processing*, 54(10):4004–4014, Oct. 2006.
- [46] B. Girod and S. Han. Optimum Update for Motion-Compensated Lifting. *IEEE Signal Processing Letters*, 12(2):150–153, Feb. 2005.
- [47] R. Xiong, F. Wu, J. Xu, S. Li, and Y.-Q. Zhang. Barbell Lifting Wavelet Transform for Highly Scalable Video Coding. In *Proc. Int. Picture Coding Symp. PCS '04*, San Francisco, USA, Dec. 2004.
- [48] H.-W. Park and H.-S. Kim. Motion Estimation Using Low-Band-Shift Method for Wavelet-Based Moving-Picture Coding. *IEEE Trans. Image Processing*, 9(4):577–587, Apr. 2000.
- [49] X. Li. New Results of Phase Shifting in the Wavelet Space. *IEEE Signal Processing Letters*, 10(7):193–195, Jul. 2003.
- [50] Y. Andreopoulos, A. Munteanu, G. Van der Auwera, J. P. H. Cornelis, and P. Schelkens. Complete-to-Overcomplete Discrete Wavelet Transforms: Theory and Applications. *IEEE Trans. Signal Processing*, 53(4):1398–1412, Apr. 2005.
- [51] Y. Andreopoulos, M. Van Der Schaar, A. Munteanu, J. Barbarien, P. Schelkens, and J. Cornelis. Complete-to-Overcomplete Discrete Wavelet Transforms for Fully-Scalable Video Coding with MCTF. *Proc. of SPIE*, 5150:719–731, Jun. 2003.
- [52] X. Li. Scalable Video Compression via Overcomplete Motion Compensated Wavelet Coding. *Signal Processing:Image Communication*, 19(7):637–651, 2004.
- [53] R. Xiong, J. Xu, F. Wu, and S. Li. Studies on Spatial Scalable Frameworks for Motion Aligned 3D Wavelet Video Coding. In *Proc. of SPIE*, volume 5960, Jul. 2005.
- [54] N Mehrseresht and D. Taubman. A Flexible Structure for Fully Scalable Motion-Compensated 3-D DWT With Emphasis on the Impact of Spatial Scalability. *IEEE Trans. Image Processing*, 15(3):740–753, Mar. 2006.
- [55] R. Xiong, J. Xu, F. Wu, S. Li, and Y.-Q. Zhang. Spatial Scalability in 3D Wavelet Coding with Spatial Domain MCTF Encoder. In *Proc. Int. Picture Coding Symp. (PCS) '04*, San Francisco, USA, Dec. 2004.
- [56] <ftp://ftp.tnt.uni-hannover.de>.
- [57] S. D. Servetto, K. Ramchandran, and M. T. Orchard. Image Coding Based on a Morphological Representation of Wavelet Data. *IEEE Trans. Image Processing*, 8(9):1161–1174, Sep. 1999.

- [58] I. E. G. Richardson. *H.264 and MPEG-4 Video Compression: Video Coding for Next-generation Multimedia*. John Wiley & Sons, 2003.
- [59] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the H.264/AVC Video Coding Standard. *IEEE Trans. Circ. Syst. Video Tech.*, 13(7):560–576, Jul. 2003.
- [60] H.-M. Wong, O. C. Au, A. Chang, S.-K. Yip, and C.-W. Ho. Fast Mode Decision and Motion Estimation for H.264 (FMDME). In *Proc. IEEE Int. Symp. on Circuits and Systems (ISCAS)*, pages 473–476, May 2006.
- [61] P. Yin, H.-Y. Cheong, A. M. Tourapis, and J. Boyce. Fast Mode Decision and Motion Estimation for JVT/H.264. In *Proc. IEEE Int. Conf. Image Processing*, pages 853–856, Sep. 2003.
- [62] Wavelet Codec Reference Document and Software Manual. *ISO/IEC JTC1/SC29/WG11, N7334*, Jul. 2005.
- [63] P. A. Chou, T. Lookabaugh, and R. M. Gray. Optimal Pruning with Applications to Tree-Structured Source Coding and Modeling. *IEEE Trans. Inform. Theory*, 35(2):299–315, Mar. 1989.
- [64] M. H. Chan, Y. B. Yu, and A. G. Constantinides. Variable Size Block Matching Motion Compensation with Applications to Video Coding. *IEE Proceedings I: Communications, Speech and Vision*, 137(4):205–212, Aug. 1990.
- [65] K. Sayood. *Introduction to Data Compression*. Morgan Kaufmann Publishers, San Francisco, 2 edition, 2000.
- [66] Z. Wu and M. He. Wavelet Image Coding by Dilation-run Algorithm. In *Proc. IEEE Int. Conf. Image Processing*, pages 3165–3168, Oct. 2004.
- [67] I. H. Witten, R. M. Neal, and J. G. Cleary. Arithmetic Coding For Data Compression. *Communications of the ACM*, 30(6):520–540, Jun. 1987.
- [68] S. Todd, G. Langdon, and J. Rissanen. Parameter Reduction and Context Selection for Compression of Gray-Scale Images. *IBM Journal of Research and Development*, 29:188–193, Mar. 1985.
- [69] Information Technology - Digital Compression and Coding of Continuous-tone Still Images - Part 1: Requirements and Guidelines. *ISO/IEC International Standard 10918-1, ITU-T Rec. T.81*, 1993.
- [70] G. K. Wallace. The JPEG Still Picture Compression Standard. *IEEE Trans. on Consumer Electronics*, 38(1):xviii–xxxiv, Feb. 1992.
- [71] S.-T. Hsiang. *Highly Scalable Subband/Wavelet Image And Video Coding*. PhD thesis, Rensselaer Polytechnic Institute, Troy, New York, 2002.

- [72] A. Islam and W. A. Pearlman. An Embedded and Efficient Low-Complexity Hierarchical Image Coder. In *Visual Communications and Image Processing '99, Proc. of SPIE*, volume 3653, pages 294–305, Jan. 1999.
- [73] N. Ahmed, T. Natarajan, and K. R. Rao. Discrete Cosine Transform. *IEEE Trans. Computers*, C-23(1):90–93, Jan. 1974.
- [74] F. A. Kamangar and K. R. Rao. Fast Algorithms for the 2-D Discrete Cosine Transform. *IEEE Trans. Computers*, C-31(9):899 – 906, Sep. 1982.
- [75] E. Feig and S. Winograd. Fast algorithms for The Discrete Cosine Transform. *IEEE Trans. Signal Processing*, 40(9):2174–2193, Sep. 1992.
- [76] K. R. Rao and P. Yip. *Discrete Cosine Transform: Algorithms, Advantages, Applications*. Academic Press, Boston, 1990.
- [77] Z. Xiong, K. Ramchandran, and M. T. Orchard. Space-Frequency Quantization for Wavelet Image Coding. *IEEE Trans. Image Processing*, 6(5):677–693, May 1997.
- [78] Maragos, P. and Schafer, R. W. Morphological Systems for Multidimensional Signal Processing. *Proc. of the IEEE*, 78(4):690–710, Apr. 1990.