Protection of Scalable Video by Encryption and Watermarking

Zafar Shahid

UNIVERSITÉ DE MONTPELLIER II (UM2)

ÉCOLE DOCTORALE I2S
INFORMATION, STRUCTURE ET SYSTEME

P h D T H E S I S

as a partial fulfillment of the requirements of the

PhD Degree in Science

by the University of Montpellier II

Speciality: Computer Science

Presented and Publicly Defended by

Zafar SHAHID

Protection of Scalable Video by Encryption and Watermarking

Protection des Vidéos Hiérarchiques par Cryptage et Tatouage

Thesis Supervisor: William Puech
Co-supervisor: Marc Chaumont

Realized as a member of the Research Team ICAR at LIRMM

defended on October 08, 2010

Jury:

Reviewers:	Atilla Baskurt	-	LIRIS, Lyon
	Claude Labit	-	INRIA, Rennes
Supervisor:	William Puech	-	LIRMM, Montpellier
Co-supervisor:	Marc Chaumont	-	LIRMM, Montpellier
President:	Alain Jean-Marie	-	LIRMM, Montpellier
Examiner:	Jacques Blanc-Talon	-	DGA, Paris

Acknowledgment

The harder I work , the more lucky I seem to be.

(Thomas Jefferson 1743-1826)

I am really thankful to all the members of my jury and it is a great honor for me to have such world over recognized experts in my jury. I am especially grateful to my reviewers Atilla Baskurt and Claude Labit whose comments and opinions have helped me to clarify and improve the quality of my manuscript.

A PhD is far from being a personal work and its fulfillment requires the support and collaboration of many people. I express my gratitude to my supervisors William Puech and Marc Chaumont who gave me the opportunity to join this project in LIRMM where I got fruitful research environment. I thank William Puech for his excellent guidance, helpful technical advices, encouragement, patience and hospitality during our collaboration that helped me through .He has always been available round the clock to discuss the rising issues. I would like to thank Marc Chaumont, in particular, for his personal warmth, insightful comments and technical discussions that helped to expose the loop holes in my work and his valuable assistance for maturing my theoretical foundation. My gratitude is also extended to other members of ICAR team. Olivier Strauss here deserves special thanks for his candid comments and interest in our presentations during team meetings. His comments helped me and other graduate students to have more clear understanding of their topics.

After my arrival in France for Master and PhD studies, the people who helped and guided me a lot were Florent Dupont and Atilla Baskurt. They helped me a lot throughout my Master studies. Florent has especially indebted permanent remarks on my career. He helped me a lot to clear the foundation of image processing and to get expert in software tools like Matlab. Even after completion of Master program, he always welcomed me whenever I visited him in LIRIS. He is not only a good researcher but also a good human being. I am also grateful to Alain Jean-Marie who accepted to be part of my PhD jury. He has also been part of CST juries during my PhD. His graduate course of *Advance Markov Modeling* really helped me to get comfortable with this subject. Thanks are also extended to all the researchers whom I met during conferences. Their opinions and feedback have been the key elements for improving my work.

My extreme gratitude goes to my colleagues. I am very fortunate to share my workplace with such knowledgeable and supportive colleagues. When I think of the ICAR Hall the names that flash through my mind are Dalila, Azhar, Bilal, Roseline, Benois, Auréline, Baptiste, Nicolas, Loïc and Alain. Administrative staff has always been supportive for all my visits to conferences and summer schools. I will never forget the smiling face of Cécile Lukasik and Nicole, whenever I visited them for some paper work regarding my administrative issues and official visits.

During my graduate studies, the support of my family has always been there. It is sad that my mother is not alive now to see me becoming a PhD. She left this world

during second year of my PhD. It was only because of Dr. Nabeel Bracha that I succeeded in getting a flight next morning and have been able to attend her funeral. The only regret which I may have is that I could not have been there with her when she was on bed before her death. My father has been and is always supportive of my PhD.

This manuscript is made possible just because of the unconditional help of my wife Ammara, who was always there to review my manuscript and gave lot of suggestions to make it better. She has scarified a lot during the final year of my PhD; first because of delay in family visa and then my preoccupied schedule. She provided me the inspiration and motivation as we experienced the thick and thin of graduate student life together. Even in times of crisis, I could always count on her to bring a smile to my face.

One of the precious things which I have got during my stay here in France is a good bunch of friends. During my master studies in Lyon, I got a chance to meet Farhan Cyprian, Asif Memon, Dr. Nabeel Bracha among others. Discussions with them were always full of knowledge and excitement. During my PhD, I am lucky to have friends like Rashid, Mubashir, Faisal and Moeez in Montpellier. I could not achieve all this without their unconditional support during this time.

Afternoon tea breaks have been the hall mark of my stay in LIRMM. Exciting discussions were always present during those leisure moments. They always served as a refreshing moment, taking me away from the woes and worries of my research. The comments of Mehdi, Asad and Naveed along with the insightful discussion with Dr. Khizar Hayat and Dr. Khalid Saleem always helped us to understand the things better.

In writing acknowledgment, my greatest fear is that I will fail to adequately convey my thanks to everyone who helped me through my work. But at the same time, I hope that I have succeeded in expressing my regards to most of the people. In the end, I would like to thank all my colleagues and the staff members of LIRMM.

Grant: I gratefully acknowledge the funding provided by ANR (*Agence Nationale de la Recherche*) and the region of Languedoc Roussillon of France for national project VOODDO (2008-2011), that supported my research at LIRMM, University of Montpellier II.

October 2010 Zafar SHAHID

Contents

List of Figures

List of Tables

Glossary

Inter frame	An *inter* frame is predicted from previous or future reference frames and only residual is encoded then, 14
Intra frame	An *intra* frame is encoded independently with only spatial prediction, 14
AVS	Audio Video Coding Standard is video codec of China, 27
C2DVLC	Huffman coding based 2D entropy coding module available in *Jizhun* profile of AVS, 27
CABAC	Arithmetic coding based entropy coding module available in *main* profile of H.264/AVC, 19
CAVLC	Huffman coding based entropy coding module available in *baseline* profile of H.264/AVC, 19
DWTSB	Discrete wavelet transform/subband coding, 152
DWTSB-AS	Discrete wavelet transform/subband with adaptive scan, 153
ELs	Enhancement layers, 152
Interlaced	A video frame whose odd and even lines are scanned separately, 15
MB	macroblocks, a video frame is encoded in the form of blocks of 16x16 pixels called macroblocks, 14
NZ	Non-zero quantized transform coefficient, 17
Progressive	A video frame in which all the lines are scanned in a single sequence, 15
ROI	Region of interest., 15
SD resolution	A video frame having resolution of 704×480, 15
SE	Selective encryption, 118

SE-C2DVLC	Selective encryption in C2DVLC entropy coding module of AVS, 141
SE-CABAC	Selective encryption in CABAC entropy coding module of H.264/AVC, 118
SE-CAVLC	Selective encryption in CAVLC entropy coding module of H.264/AVC, 118
SSE	Selective and scalable encryption, 152
Trailing ones	In CAVLC, QTCS in right-bottom corner of DCT block and with magnitude '1' are coded as separate syntax element called trailing ones, 24

Chapter 1

Introduction

Le champ du traitement des images et des vidéos attire l'attention depuis les deux dernières décennies. Ce champ couvre maintenant un spectre énorme d'applications comme la TV 3D, la télé-surveillance, la vision par ordinateur, l'imagerie médicale, la compression, la transmission, etc. En ce début de vingt et unième siècle nous sommes témoins d'une révolution importante. Les largeurs de bande des réseaux, les capacités de mémoire et les capacités de calcul ont été fortement augmentés durant cette période. Un client peut avoir un débit de plus de 100 mbps tandis qu'un autre peut utiliser une ligne à 56 kbps. Simultanément, un client peut avoir un poste de travail puissant, tandis que d'autres peuvent avoir juste un téléphone mobile. Au milieu de ces extrêmes, il y a des milliers de clients avec des capacités et des besoins très variables. De plus, les préférences d'un client doivent s'adapter à sa capacité, par exemple un client handicapé par sa largeur de bande peut être plus intéressé par une visualisation en temps réel **sans interruption** *que d'avoir une haute résolution. Pour y faire face, des architectures hiérarchiques de codeurs vidéo ont été intoduites afin de comprimer une seule fois, et de décomprimer de différentes manières. Comme la DCT n'a pas la fonctionnalité de multi-résolution, une architecture vidéo hiérarchique est conçue pour faire face aux défis des largeurs de bande et des puissances de traitement hétérogènes.*

Avec l'inondation des contenus numériques, qui peuvent être facilement copiés et modifiés, le besoin de la protection des contenus vidéo a pris plus d'importance. La protection de vidéos peut être réalisée avec l'aide de trois technologies : le tatouage de méta-données et l'insertion de droits d'auteur, le cryptage pour limiter l'accès aux personnes autorisées et la prise des empreintes digitales active pour le traçage de traître. L'idée principale dans notre travail est de développer des technologies de protection transparentes à l'utilisateur. Cela doit aboutir ainsi à un codeur vidéo modifié qui sera capable de coder et d'avoir un flux de données protégé. Puisque le contenu multimédia hiérarchique a déjà commencé à voir le jour, il y a un besoin d'algorithmes pour la protection indépendante des couches d'amélioration.

Ce manuscrit est concentré sur les protections de flux de vidéos de pointe et leurs extensions hiérarchiques. Au début de notre travail, nous optons pour l'insertion de messages avec forte capacité avec un compromis très contrôlé entre le PSNR et le débit. Le secret se trouve dans le fait que l'on prenne en compte la boucle de reconstruction pour éviter la disparité entre l'encodeur et le décodeur. La comparaison a été faite avec des techniques de tatouage plus naïves afin de prouver sa supériorité. Le tatouage par étalement de spectres est la technique mise en oeuvre sur les coefficients transformés quantifiés à l'intérieur de la boucle de reconstruction.

Cette même méthode est, plus tard, utilisée pour insérer des codes de Tardos pour le traçage de traître.

Dans la deuxième partie, nous proposons une méthode de cryptage sélectif pour les vidéos H.264/AVC et AVS afin de limiter l'accès aux utilisateurs autorisés. Pour cela, le module de codage entropique a été modifié afin de le transformer en un module de crypto-compression, qui peut exécuter le cryptage et la compression en même temps. Le flux de données chiffré respecte entièrement le format original et conserve exactement le même débit. De plus, la puissance de traitement supplémentaire exigée pour le codage et le décodage du flux de données chiffré est très négligeable. Notre algorithme de cryptage sélectif pour le codeur entropique CABAC est le premier algorithme de cryptage sélectif pour un codeur arithmétique. Une analyse détaillée de la sécurité a été détaillée afin de vérifier les méthodes proposées. Après le cryptage sélectif, le centre de nos recherche s'est orienté vers les vidéos hiérarchiques. Nous avons proposé une méthode de parcours hiérarchique des couches d'amélioration des vidéos hiérarchiques qui garantit une meilleure compression sans compromis sur le PSNR et l'efficacité. Des algorithmes pour la protection indépendante des couches d'amélioration dans une vidéo hiérarchique a aussi été proposé dans ce travail.

Ce manuscrit est organisé en deux parties. La partie I illustre les concepts théoriques nécessaire pour une meilleure compréhension de notre travail. Nous présentons une vue d'ensemble des codeurs vidéos, en particulier, H.264/AVC et AVS (chapitre 2), tatouage (chapitre 3), cryptage sélectif (chapitre 4) et empreintes digitales actives (chapitre 5) pour les images et les vidéos.

Nos contributions sont présentées dans la partie II. Dans le chapitre 6, nous présentons l'insertion de messages et d'empreintes digitales dans des vidéos H.264/AVC durant le processus de codage. Cela inclut des solutions avec de grandes capacités pour des applications de méta-données, et des techniques robustes de tatouage pour l'insertion de droits d'auteur ou d'empreintes digitales. Dans le chapitre 7, nous présentons des techniques de cryptage sélectif pour les principaux modes du codeur H.264/AVC et pour le mode de Jizhun pour AVS. Le chapitre 8 se concentre sur des cryptages sélectifs de vidéos hiérarchiques. Il contient des méthodes pour protéger les couches d'amélioration de vidéos hiérarchiques en utilisant le mélange des parcours tout en conservant le débit initial et en utilisant le minimum de ressources supplémentaires. Ce manuscrit se termine par le chapitre 9 autour d'une discussion de tous les résultats obtenus ainsi que des perspectives.

Field of image and video processing has got lot of attention during the last two decades. This field now covers a vast spectrum of applications like 3D TV, tele-surveillance, computer vision, medical imaging, compression, transmission and much more. Of particular interest is the revolution being witnessed by the first decade of twenty-first century. Network bandwidths, memory capacities and computing efficiencies have got revolutionized during this period. One client may have a 100 mbps connection whereas the other may be using a 56 kbps dial up modem. Simultaneously, one client may have a powerful workstation while others may have just a smart-phone. In between these extremes, there may be thousands of clients

with varying capabilities and needs. Moreover, the preferences of a client may adapt to his capacity, *e.g.* a client handicapped by bandwidth may be more interested in real-time visualization **without interruption** than in high resolution. To cope with it, scalable architectures of video codecs have been introduced to 'compress once, decompress many ways' paradigm. Since DCT lacks the multi-resolution functionality, a scalable video architecture is designed to cope with challenges of heterogeneous nature of bandwidth and processing power.

With the inundation of digital content, which can be easily copied and modified, the need for protection of video content has got attention. Video protection can be materialized with help of three technologies: watermarking for meta data and copyright insertion, encryption to restrict access to authorized persons, and active fingerprinting for traitor tracing. The main idea in our work is to make the protection technology transparent to the user. This would thus result in a modified video codec which will be capable of encoding and playing a protected bitstream. Since scalable multimedia content has already started coming to the market, there is a need for algorithms for independent protection of enhancement layers.

This manuscript is focused on protections of state of the art video bitstreams and their scalable extensions. In the beginning of our work, we opt for watermark embedding for high payload with very controlled compromise on PSNR and bitrate. The secret lies in the fact that the reconstruction loop has taken into account to avoid the mismatch on the encoder and decoder. The comparison was also made with naive watermarking techniques to prove its superiority. Spread spectrum watermarking technique was then implemented on quantized transformed coefficients inside the reconstruction loop. It is, later on, used to embed Tardos fingerprinting code to avoid traitor tracing.

In the second part, We propose a scheme for selective encryption of H.264/AVC and AVS video content to restrict the access to only authorized users. For this purpose, the entropy coding module has been modified to transform a video codec into a crypto-compression module, which can perform the encryption and compression at the same time. The encrypted bitstreams are fully format compliant with exactly the same bitrate. In addition, the extra processing power required for encoding and decoding encrypted bitstreams is very negligible. Our selective encryption algorithm for CABAC entropy coding scheme is the first format compliant SE algorithm for an arithmetic entropy coder. A detailed security analysis has been performed for the verification of the proposed scheme. Once a basic selective encryption is in place, the focus is shifted to scalable video. A method for adaptive scan of enhancement layers of scalable video has been proposed which guarantees extra compression without compromise on PSNR and efficiency. Algorithms for independent protection of enhancement layers in a scalable video have also been proposed in this work.

This manuscript is organized into two main parts. Part I illustrates the theoretical concepts needed for the better understanding of our work. We present architecture of a basic video codec along with an overview of H.264/AVC and AVS (Chapter 2), watermarking (Chapter 3), selective encryption (Chapter 4) and active fingerprinting (Chapter 5) of image/video data.

Our contributions are being presented in Part II. In Chapter 6, we present the message embedding and active fingerprinting of H.264/AVC video during the video encoding process. It includes both high payload solutions for metadata applications, and robust watermarking techniques for insertion of copyright/fingerprinting code. In Chapter 7, we present the selective encryption techniques for base & main profile of H.264/AVC and Jizhun profile of AVS. Chapter 8 focuses on issues related to selective encryption of scalable video. It contains techniques to protect the enhancement layers of scalable video using scrambling of scan patterns while preserving the bitrate and utilizing minimal extra resources. The manuscript is concluded in Chapter 9 by discussion of overall results and presentation of the future prospects.

Part I

STATE OF THE ART

Dans la première partie de ce manuscrit, nous présentons les concepts fondamentaux nécessaires pour comprendre nos travaux de recherche. Dans le chapitre 2, l'architecture de codage vidéo est introduite, suivie par un état de l'art des standards de codage vidéo H.264/AVC et AVS (Audio Video Standard pour la Chine). Pour H.264/AVC, les points particuliers présentés concernent les modules de transformation et de codage entropique, qui seront modifiés pour le tatouage et le cryptage de vidéos. Pour AVS, le fonctionnement du module de codage entropique est également détaillé. L'introduction de métriques objective et subjective de qualité sont également présentées dans ce chapitre. Dans le chapitre 3, les méthodes standards de tatouage sont présentées jusqu'aux travaux de recherche récents en tatouage de d'images et de vidéos. Le cryptage d'images et de vidéos pour la confidentialité et l'accès restreint est présenté dans le chapitre 4. Le dernier chapitre de la partie I, chapitre 5, présente les differents aspects des codes d'empreinte digitales qui sont utilisés pour le traçage de traître. Dans ce contexte, nous attacherons une attention spéciale aux codes d'anti-collusion. Les chapitres inclus dans cette partie serviront de base théorique au travail de recherche présenté dans la partie II.

In the first part of this manuscript, we are presenting the fundamental concepts needed to understand our research work. In Chapter 2, video codec architecture is being introduced, followed by state of the art video codec standard H.264/AVC and AVS (Audio Video Standard) of China. In H.264/AVC, special focus is given to transform coding and entropy coding modules, which will be modified for video watermarking and encryption. In AVS, working of entropy coding module is detailed. Introduction to objective and subjective quality metrics are also presented in this chapter. In Chapter 3, basic introduction of watermarking is presented along with resent research work for watermarking of image and video data. Encryption of image and video content for confidentiality and restricted access is presented in Chapter 4. The last chapter of Part I, Chapter 5, presents various aspects of fingerprinting codes which are used for traitor tracing. In this context, special attention is being given to anti-collusion codes. The chapters included in this part would serve as a theoretical foundation to the practical work being presented in the Part II.

Chapter 2

Video Codec Design and H.264/AVC

Contents

2.1 Introduction

L'objectif principal de ce chapitre est de présenter l'architecture de codage vidéo basée sur la structure de compensation de mouvement hybride. La structure de compensation de mouvement hybride est l'architecture la plus largement acceptée pour les codeurs vidéo de pointe. En commençant par le codage de source, nous

développons progressivement l'architecture d'un codeur vidéo moderne. Après l'introduction sur la conception de codeur vidéo, le point principal concerne les codeurs vidéo bien connus incluant H.264/AVC et AVS. Finalement, l'architecture vidéo hiérarchique est présentée en mettant un accent sur l'adaptabilité spatiale. Des métriques de qualité objectives et subjectives, qui seront utilisées pour l'analyse expérimentale de notre travail, sont aussi présentées dans ce chapitre.

The main objective of this chapter is to present the architecture of video codec based on motion compensated hybrid framework. Motion compensated hybrid framework is the most widely accepted architecture of state of the art video codecs. Starting from source coding, we gradually develop the architecture of a modern video codec. After the introduction of video codec design, the focus is then shifted to well-known video codecs, which include H.264/AVC and AVS. Finally, scalable video architecture is presented with an emphasis on the spatial scalability.

This chapter is arranged as follows. Section 2.2 presents the basic building blocks of a modern video codec. Architecture of state of the art video codec H.264/AVC is presented in Section 2.3, while details of AVS are presented in Section 2.4. Section 2.5 presents the scalable architecture of H.264/AVC. For the sake of the comparison, we are putting forward some well known quality metrics in Section 2.6, where as Section 2.7 concludes the chapter.

2.2 Video codec design

Video data contains three types of redundancy. Neighboring pixels in the same video frame are correlated, and hence video data has **spatial redundancy**. Neighboring video frames are very similar to each other, so it has **temporal redundancy**. Third type of redundancy present in video data is **statistical redundancy**. A video codec compresses the video signal by minimizing all three types of redundancies. In this section, we are going to present a basic video compression system, starting from where it historically evolved.

2.2.1 Basic compression system

Entropy is defined as the average information content of the source X. For a source symbol i, with probability $P(i)$, the 1^{st} order entropy $H(X)$ is defined as:

$$H(X) = -\sum_{i=0}^{m-1} P(i) log_2(P(i)) bits. \tag{2.1}$$

In the 1960s, the basic compression system was an entropy coding module, which removes redundancy and increases the entropy of data. In the simplest form, the entropy coding represents the frequent symbols by smaller number of bits, while using larger string of bits for less frequent symbols. The compression ratio depends on using symbols in accordance with the histogram. If the smaller sequences are used for

less frequent symbols, the file size may increase. Huffman coding [Huffman 1952], run-length coding and dictionary based coding methods [Ziv 1977] are well known examples of entropy coding systems. This family was joined by arithmetic coding [Moffat 1998] later on. The block diagram of such type of compression system is shown in Fig. 2.1.

The compression ratio in this case depends on the sparseness of the input data. For example, data with super-Gaussian histogram (Fig. 2.2.a) will be compressed more than data having sub-Gaussian histogram (Fig. 2.2.b).

Figure 2.1: Basic compression system.

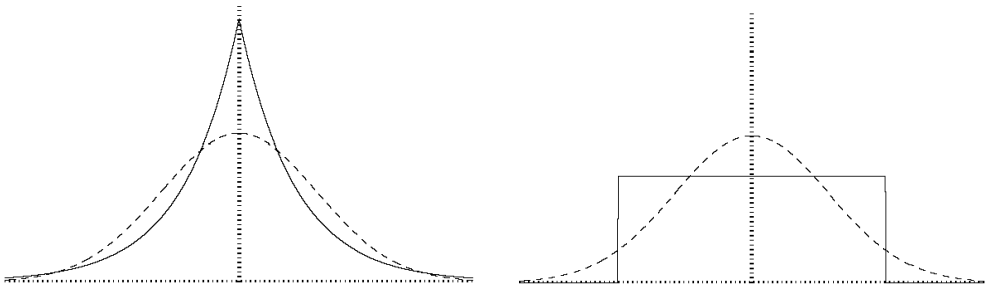


Figure 2.2: Examples of histogram distributions: (a) Super-Gaussian distribution, (b) Sub-Gaussian distribution.

2.2.2 Transform coding

Common video sources have lot of redundancy that simply entropy coding cannot remove. In transform coding the data is transformed to some other space where it has sparse representation having high peak and heavy tails. Hence, entropy coding module can compress it more efficiently as shown in Fig. 2.3. For video data, discrete cosine transform (DCT) and discrete wavelet transform (DWT) are most widely used transforms. Video codecs work on a block of 16×16 called a marcroblock (MB). Fig. 2.4 presents working of DCT with the help of an example.

Frequency and time informations are complementary in cosine transform. For example, pulse in time domain is constant in frequency domain and pulse in frequency domain is infinite sinusoidal in time domain. But most real life signals are limited both in space and frequency. The main limitation of DCT is that it cannot perform well at discontinuities and edges and can decorrelate data up to 2nd order only. Moreover it does not perform well in case of non-Gaussian data [Goyal 2001]. That is why context-modeling is also used in all state of the art image and video codecs.

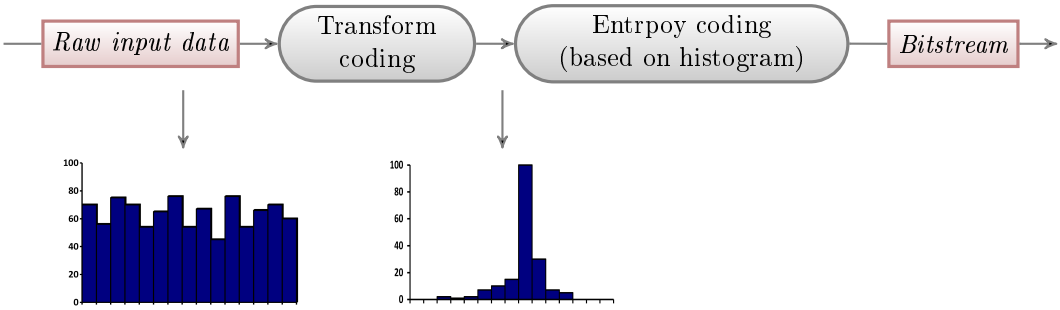


Figure 2.3: Basic compression system along with transform coding.

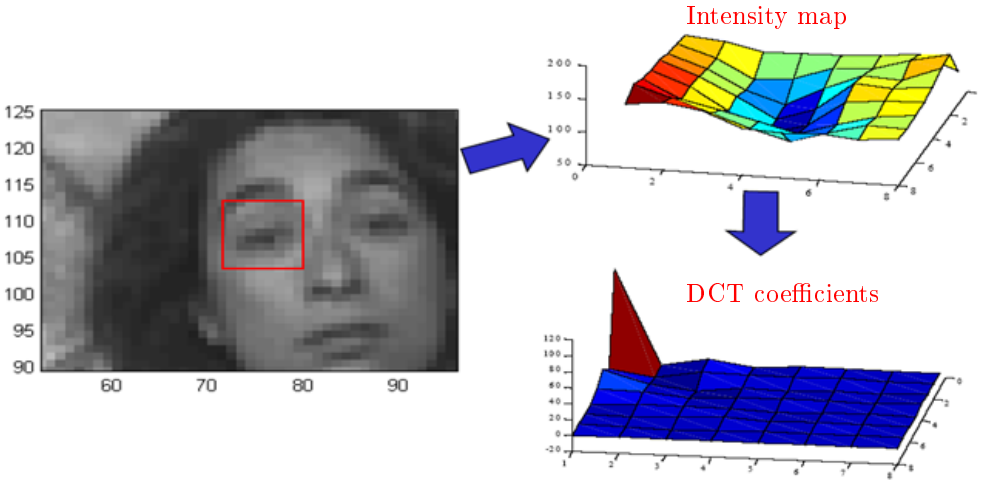


Figure 2.4: Compression efficiency of DCT transform for image data.

2.2.3 Quantization

This compression system is lossless and is required in applications where exact reconstruction is required e.g. text data, but the compression ratio is very less (up to 1:2 only). Lossy compression achieves better compression ratio and multimedia data can be compressed with lossy compression as shown in Fig. 2.5. Quantization is used to quantify the transformed coefficients and obtain compression gain at the expense of quality. Mathematically, the output signal of quantization step can be described as:

$$c_O = \lfloor |c_I| / \Delta \rfloor \, sign(c_I), \tag{2.2}$$

where c_I and c_O are the input and output coefficients, respectively. This step involves information loss. For lossless coding, Δ must be set to unity.

Quantization may be uniform or non-uniform. In uniform quantization, the quantization step size is the same for all the steps, while it is different in non-uniform quantization (*e.g.* logarithmic quantization in H.264/AVC) as shown in Fig. 2.6. In previous DCT based codecs, quantization matrices have also been used *e.g.*, JPEG

quantization matrices. Quantization in H.264/AVC works differently from that of MPEG-1/2/4 and is on a logarithmic scale. The mapping is approximately:

$$QP_{H.264} = 12 + 6 * log2(QP_{MPEG}). \tag{2.3}$$

For example $QP_{MPEG} = 2$ is equivalent to $QP_{H.264} = 18$ approximately.

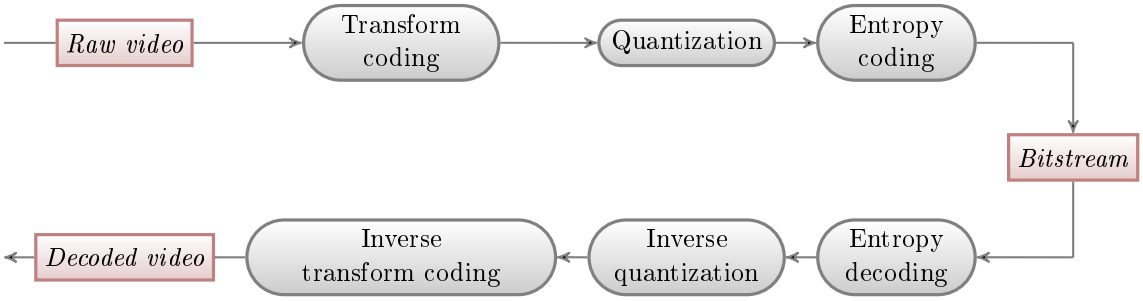


Figure 2.5: Basic compression system along with transform coding and quantization.

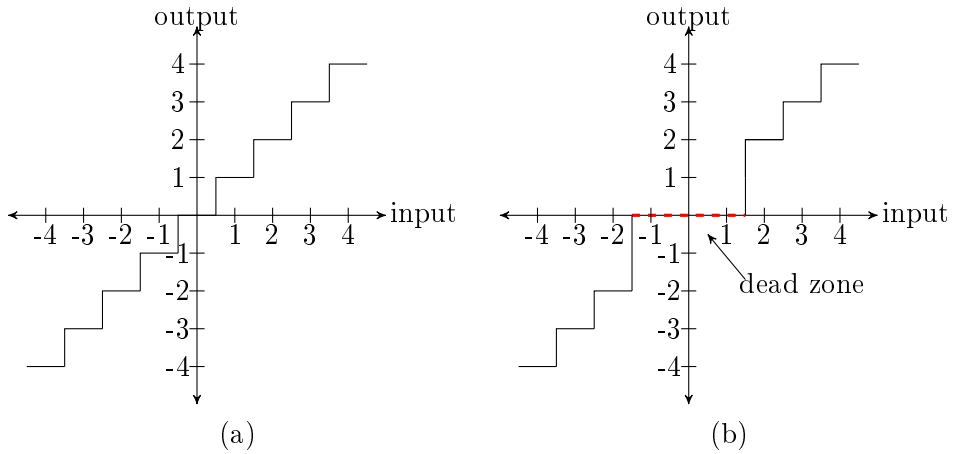


Figure 2.6: Quantization types on the basis of step size: (a)Uniform quantization, (b) Non-uniform quantization.

2.2.4 *Intra* prediction

In a video frame, spatial correlation is minimized by prediction from neighboring decoded blocks called *intra* prediction as shown in Fig. 2.7. *Intra* prediction can be performed either in spatial or in frequency domain. For example, it is performed in MPEG2 *i.e.*, DC coefficient prediction is performed in MPEG2 from top and left MBs. While in AVS video coding standard, *intra* prediction is performed in the pixel domain.

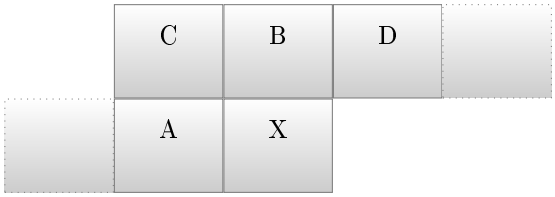


Figure 2.7: X is current block while A, B, C and D are top and left neighboring block used for *intra* prediction.

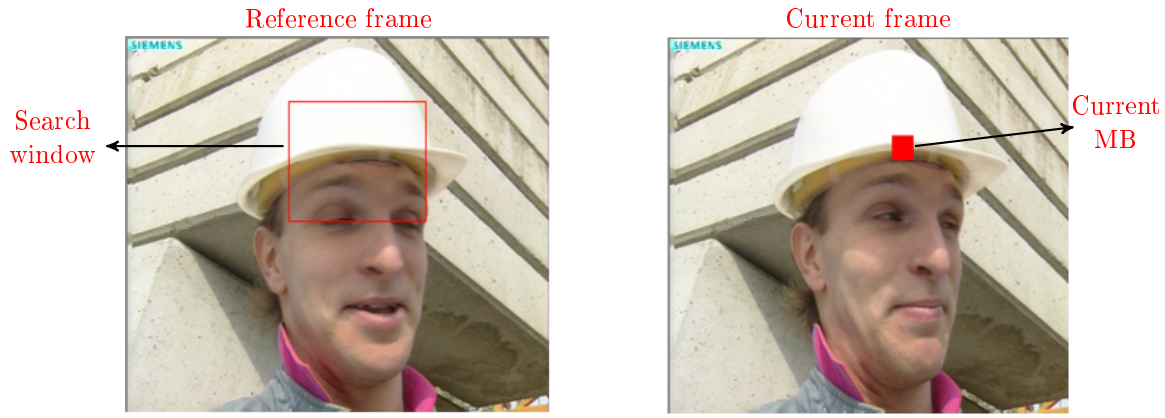


Figure 2.8: Motion estimation of a MB in current frame from previous frame.

2.2.5 Motion estimation

For entropy coding, transform coding and quantization, image and video codecs were the same. In this section, we want to remove the temporal redundancy from the video data. For this purpose, motion estimation is used as shown in Fig. 2.8. A block of pixels in the present video frame is searched and estimated from a block in another frame (temporally previous or next), and the residual is then encoded using transform coding, quantization and entropy coding modules. It is the most important part of video codec and takes up to 70 % of overall processing of video encoder.

2.2.6 Pre-processing

Raw video is pre-processed before being compressed by a video encoding module. Several operations are performed in the preprocessing stage *e.g.*, color space conversion, noise removal, resolution scaling or some control operations which are later exploited by the encoding module to encode the video effectively.

Rate control is a module which controls the bitrate while attaining the maximum quality of compressed video. It is an integral part of video codec in heterogeneous networks. For example, for a 3G wireless connection having bandwidth of 384 kbps,

we would like to have a video bitstream of exactly the same bitrate. If the bitrate is higher, it cannot be viewed over this channel in real-time. If it is lesser than this limit, we are under utilizing the bandwidth. It is the responsibility of rate control module to keep the bitrate constant. To allocate bits intelligently to different parts of video, it gets information about video content from pre-processing stage. For example, rate control is informed about a scene change, which may insert an *intra* frame to save bits, while conserving the quality. Even inside a frame, variance of MBs is analyzed to decide whether they should be encoded as *intra* or *inter* MBs.

Frame rate conversion between 24/30fps is also performed in pre-processing stage using 3:2 pull-down/pull-up mechanism. Most of the Hollywood movies are recorded for cinema having frame rate of 24fps. Hence DVDs/VCDs of these movies are encoded at 24fps, while the DVD players output the videos at 30fps for display monitors. So the frame rate is changed from 24fps to 30fps by 3:2 pull-up process.Frame rate conversion between 24/30fps is also performed in pre-processing stage using 3:2 pull-down/pull-up mechanism. Most of the Hollywood movies are recorded for cinema having frame rate of 24fps. Hence DVDs/VCDs of these movies are encoded at 24fps, while the DVD players output the videos at 30fps for display monitors. So the frame rate is changed from 24fps to 30fps by 3:2 pull-up process.While recording the movie from cable using set-top box at home, it is required to detect the original frame rate. If the original frame rate is 24fps, redundant fields are removed before compression.

Video resolution scaling and conversion from interlaced to progressive video is also performed in pre-processing stage. A conventional SD camera will always output interlaced frames of SD resolution at 30 fps. To record this video with a video encoder of VCD resolution (CIF - 352×240), input video is scaled-down from SD resolution to CIF. Conversion from interlaced to progressive video is also performed when the video is encoded as frames.
In case of low brightness scenes, there is lot of noise in the video data. Noise is removed before the video frame is sent to encoder for compression. Motion compensated temporal noise filters are used to remove noise, while taking into account the motion in the video scene.

Video feature detection and ROI detection are also performed in pre-processing stage. For example, sometimes we want to detect the type of video content *e.g.*, whether it contains texture or motion. ROI is normally used in order to compress it with better quality or to protect it using encryption.

Color conversion from RGB to YUV is also performed during the pre-processing stage. The RGB to YUV transform involves the following equation:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.16875 & -0.33126 & 0.5 \\ 0.5 & -0.41869 & -0.08131 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix}.$$

The inverse would then be:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.402 \\ 1 & -0.34413 & -0.71414 \\ 1 & -1.772 & 0 \end{bmatrix} \times \begin{bmatrix} Y \\ U \\ V \end{bmatrix}.$$

Human eye is more sensitive to *luma* component (Y) than *chroma* component. In normal video compression systems, YUV 420 format is used in which *chroma* is subsampled as shown in Fig. 2.9.

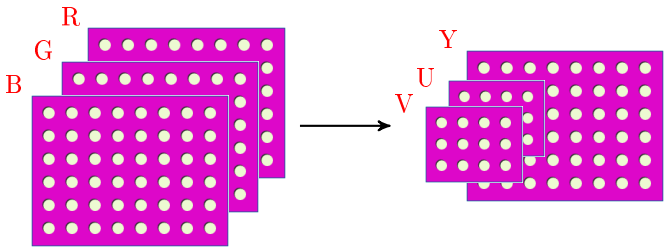


Figure 2.9: Conversion of RGB color space to YUV 420 color space with subsampled *chroma* component.

2.2.7 Post-processing

Post-processing is an important part of video compression system. Decoded video frame is normally processed before displaying it. Several tasks may be performed in post processing module e. g., adjusting the resolution of decoded video frame, removing coding artifacts etc. Some of the post processing operations are just inverse of the tasks performed in pre-processing stage. For example, spatial resolution is changed according to the display device. 3:2 pull up is performed if the decoded video has frame rate of 24 fps. Decoder provides this information to the post processing module after reading it from the frame header.

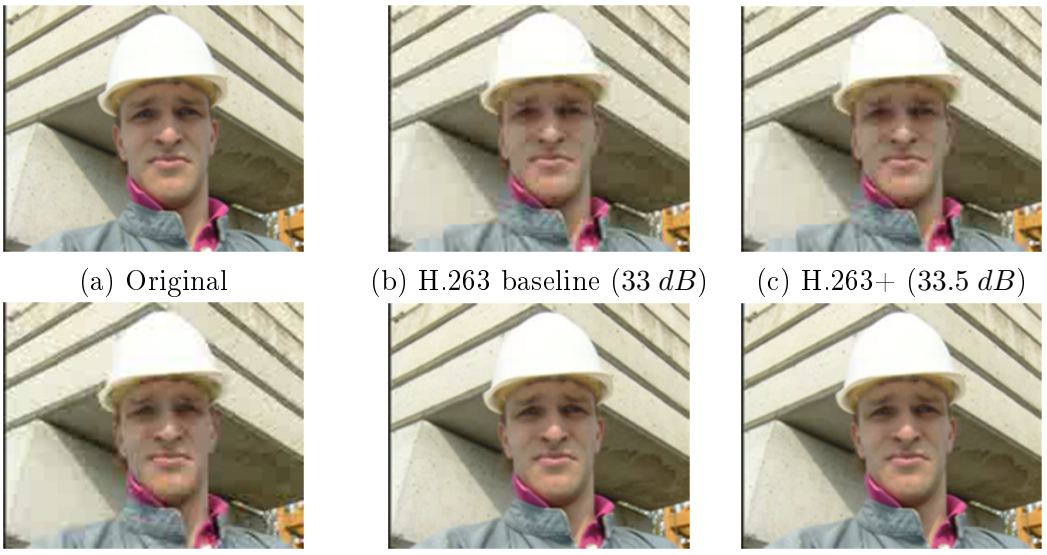
Decoded video contain video coding artifacts *e.g.*, blocking (owing to bock based video codec system) and ringing. Deblocking and deringing filters are employed to remove these artifacts. Edge enhancement filters are also used to enhance the edges in the video frames. Artificial texture is also added in post-processing module. For example, flat surfaces (without texture or motion) having low brightness, blocking artifacts are visible. To remove this problem, artificial texture is added in these areas of video frames. These areas can be detected by observing the DCT coefficients and then texture is added in these areas.

Computer monitors are progressive display devices, so we have to de-interlace the video before displaying it on such devices. This process is also performed in the post processing module. Post-processing module is hence an essential part of a video display system.

2.2.8 Modern video codec architecture

State of the art video coding standards are lot more complex and are equipped with complex tools. They perform better than previous video coding standards.

Fig. 2.10 presents a comparison of PSNR of H.264/AVC and AVS with previous standards including MPEG4 part-2. One can note that H.264/AVC and AVS outperforms previous video coding standards with difference in PSNR of more than 8 *dB* for H.264/AVC and more than 6 *dB* for AVS for QCIF resolution with 30 fps at 100 kbps. In a modern video codec, a video frame is divided into macroblocks



(a) Original (b) H.263 baseline (33 *dB*) (c) H.263+ (33.5 *dB*)

(d) MPEG4 part-2 (33.5 *dB*) (e) H.264/AVC (42 *dB*) (f) AVS (40.9 *dB*)

Figure 2.10: Comparison of H.264/AVC and AVS with previous video coding standards for QCIF resolution with 30 fps at 100 kbps.

(MBs) of 16x16 pixels and each of them is encoded separately. The block diagram of modern video codecs is shown in Fig. 2.11. Each video frame can be encoded as *intra* or *inter*. In *intra* frame, the current MB is predicted spatially from MBs which have been previously encoded and reconstructed (MB at top and left). *Inter* frame has two types of frames, namely P frames and B frames as shown in Fig. 2.12. In P frames, motion compensated prediction is done from previous reference frames. While in B frames, bi-directional motion compensated prediction is performed from both previous and future reference frames. The purpose of the reconstruction in the encoder is to ensure that both encoder and decoder use identical reference frames to create the prediction. If this is not the case, the predictions in encoder and decoder will not be identical, leading to an increasing error or "drift" between the encoder and decoder.

The difference between original and predicted frame is called residual. This residual is coded using transform coding followed by quantization and zigzag scan. In the last step, entropy coding comes into action. Non-zero coefficients (NZs) are scanned in zigzag fashion (from high frequency NZs to low frequency NZs) in the entropy coding module as shown in Fig. 2.13. On the decoding side, compressed bitstream is decoded by entropy decoding module, followed by inverse-zigzag scan. These coefficients are then rescaled and inverse transformed to get the residual signal

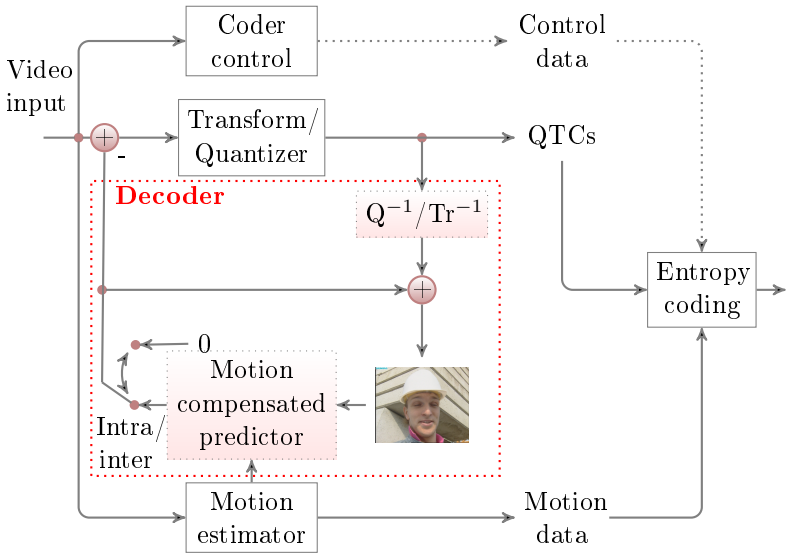


Figure 2.11: Block diagram of a modern video encoder.

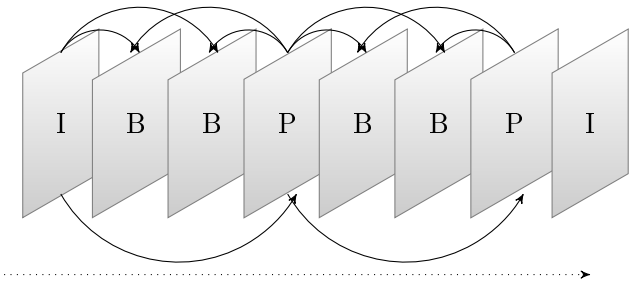


Figure 2.12: I, P and B frame types in a video sequence.

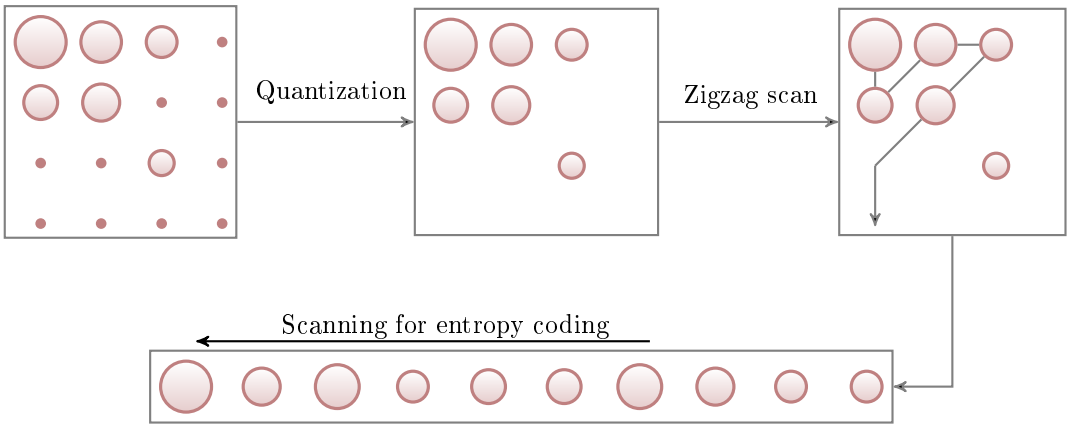


Figure 2.13: Quantization and zigzag scan of 4×4 NZs in a modern video codec.

which is added to the predicted signal to get the decoded video frame.

Modern video codecs can encode any video resolution, whose height and width are multiple of 16. But several video resolutions have got standardized for different application. For example, a low bitrate modem dial-up connection can use only low resolution video *e.g.* QCIF, while work stations can playback high definition (HF) resolutions. Fig. 2.14 illustrates several video resolutions, which have been widely accepted by industry.

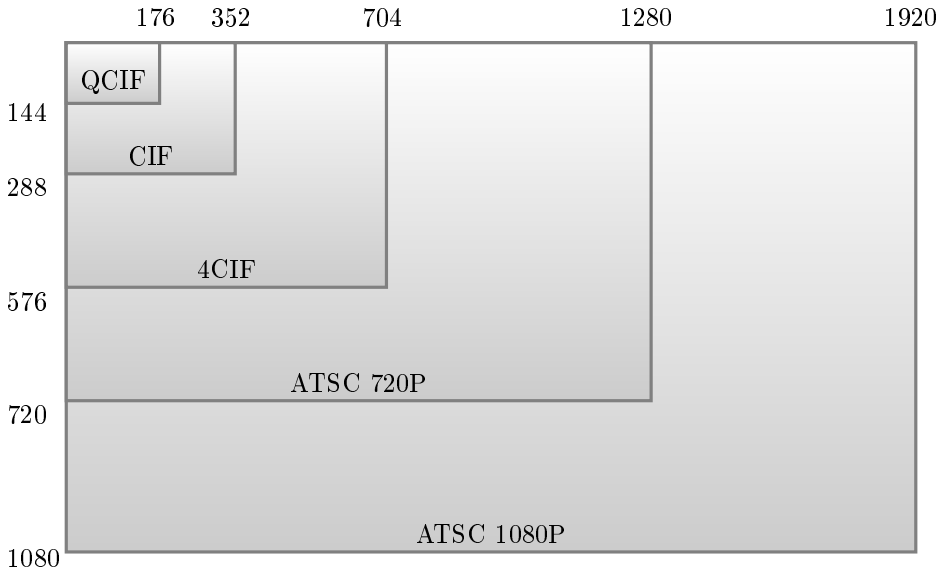


Figure 2.14: Different video resolutions targeted for different applications.

2.3 H.264/AVC

H.264/AVC (also known as MPEG4 Part 10) [H264 2003] is state of the art video coding standard of ITU-T and ISO/IEC. Being the latest and having the best compression performance, an overview of H.264/AVC is presented in Section 2.3.1. A detailed discussion of integer transform and quantization module of H.264/AVC is presented in Section 2.3.2. It is followed by an introduction of CAVLC and CABAC in Section 2.3.3 and Section 2.3.4 respectively.

2.3.1 Overview of H.264/AVC

H.264/AVC supports several profiles. *Baseline* profile of H.264/AVC is mainly for low bitrate mobile/telephony applications, while *main* profile for broadcast applications. Table 2.1 shows the available features for both profiles. Context adaptive variable length coding (CAVLC) [Bjontegaard 2002] entropy coding, which is based on Huffman coding, is supported in both profiles.

Table 2.1: *Baseline* and *main* profiles of H.264/AVC.

BASELINE	COMMON FEATURES	*MAIN*
Arbitrary Slice Ordering	Motion Prediction:7 block sizes, Quarter Pixel Accuracy, Multiple Reference Frames	B Pictures: several prediction modes
Flexible Macroblock Ordering	*Intra* Prediction:17 modes Reversible Integer Transform	Weighted Prediction
Redundant Slices	Non-uniform Quantization CAVLC Loop(deblocking)Filter	Adaptive Frame/Field Coding CABAC

Baseline profile has Arbitrary Slice Ordering (ASO) to reduce the latency in real-time communication applications, as well as the use of Flexible Macroblock Ordering (FMO) and redundant slices to improve error resilience in the coded bit stream. The *main* profile enables additional reduction in bandwidth over the *baseline* profile through mainly sophisticated Bi-directional prediction (B-pictures), Context adaptive binary arithmetic coding (CABAC) [Marpe 2003] and weighted prediction. CABAC entropy coding is also supported which is based on arithmetic coding.

H.264/AVC has some additional features as compared to previous video standards including MPEG2 and MPEG4 Part II. It has 4×4 integer transform (IT) in contrast to 8×8 transform of previous standards.

In *intra* mode, H.264/AVC has three modes, $Intra\_4 \times 4$, $Intra\_16 \times 16$ and I_PCM. In $Intra\_16 \times 16$ mode, Hadamard transform is further used to encode DC coefficients. In $Intra\_16 \times 16$ mode, entire MB is predicted from top and left neighboring pixels and has 4 modes namely *horizontal*, *vertical*, *DC* and *plane* modes. In $Intra\_4 \times 4$ mode, each 4×4 *luma* block is predicted from top and left pixels of reconstructed 4×4 neighbors and has 9 prediction modes. I_PCM mode is used to limit the maximum size of encoded block and is directly entropy encoded by bypassing the transform and quantization stages. Fig. 2.15 illustrates four of nine predictions for $Intra\_4 \times 4$ mode.

The scanning of these 4×4 blocks inside MB is not in a raster scan fashion as illustrated with the help of numbers in Fig. 2.16. In case of $Intra\_16 \times 16$ mode, Hadamard transform coefficients are sent first.

Scalar quantization with logarithmic step size has covered a wide range of bitrates. We can encode with high fidelity at lower QP, while high QP value can be used in low bitrate values. Fig. 2.17 contains an example in which for QP 10, we generate a bitstream with birate of 4.2 mbps, while the bitrate is decreased to 160 kbps for QP value of 36. An inloop deblocking filter has been provided with H.264/AVC standard. It is really a useful tool especially at higher QP values. Visual quality is enhanced with less deblocking artifacts as shown in Fig. 2.18. For *inter* frame, multiple reference frames feature has been allowed which results in extra compression in case of periodic motion. For examples, this feature is useful for periodic motions of wings of a flying bird as shown in Fig. 2.19.

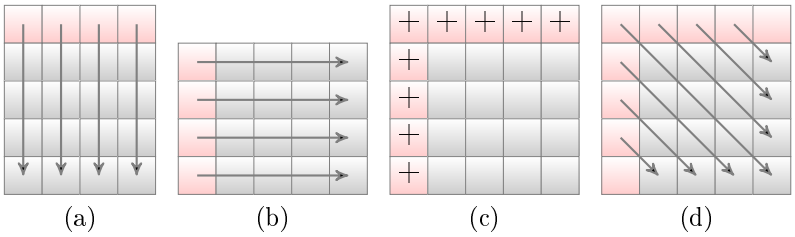


Figure 2.15: Four 4×4 *intra* prediction modes of H.264/AVC: (a) Vertical, (b) Horizontal, (c) DC, (d) Diagonal.

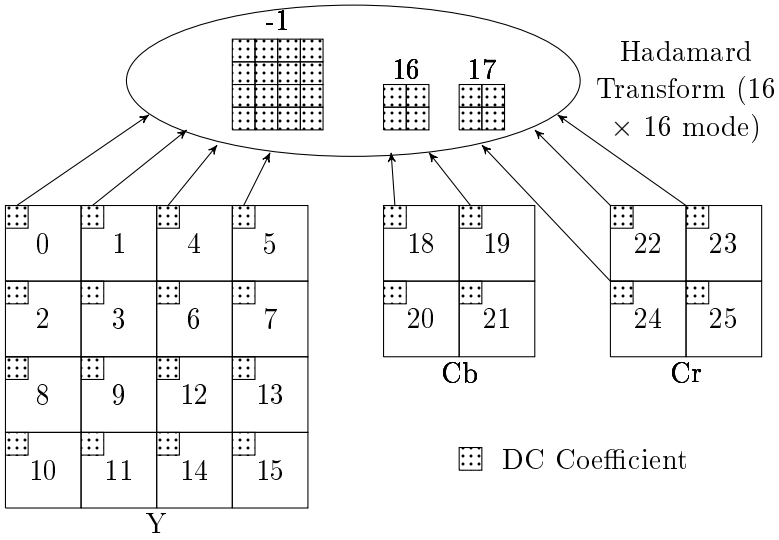
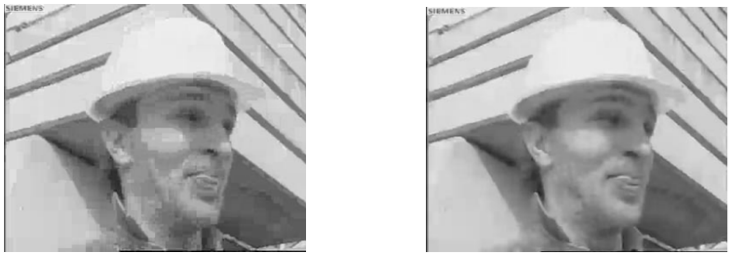


Figure 2.16: Order of transmission of $Intra\_4 \times 4$ blocks inside MB.

(a) QP = 10, Bitrate=4.2mbps (b) QP = 36, Bitrate=160kbps

Figure 2.17: Effect of quantization on PSNR and bitrate of H.264/AVC.

In addition to these feature, H.264/AVC supports several other useful features *e.g.*, variable block size motion estimation, quarter pixel accuracy, improved skipped and direct motion inference. For *intra* frame, it offers additional spatial prediction modes. All these additional features of H.264/AVC are aimed to outperform previous video coding standards [Wiegand 2003].



(a) Without deblocking filter (b) With inloop deblocking filter

Figure 2.18: Performance of inloop deblocking filter of H.264/AVC.

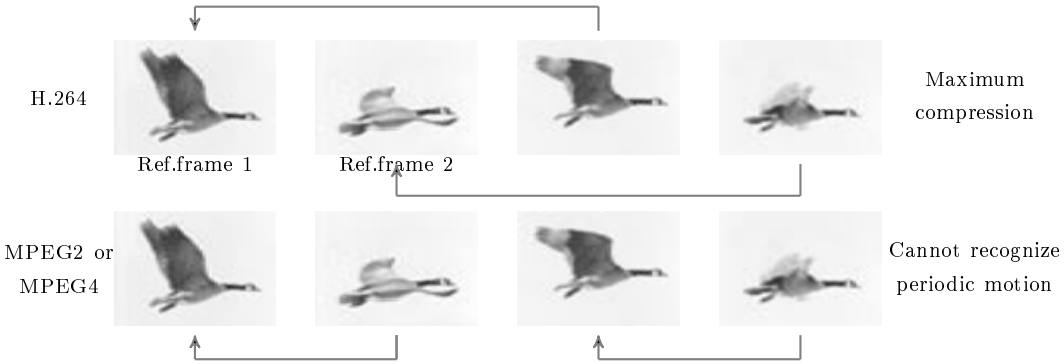


Figure 2.19: Multiple reference frame support of H.264/AVC is useful in case of periodic motion.

After presenting an overview of H.264/AVC, we present integer transform and quantization in the following section. Here, we have used capital letters to represent matrices e.g. A, Y, W and small letters along with index to represent the elements of matrices e.g. $x(i, j)$ represents j^{th} element in i^{th} row of matrix X.

2.3.2 Integer transform and quantization

In *baseline* profile of H.264/AVC, it has 4×4 IT in contrast to 8×8 transform of previous standards. In higher profiles, it offers transform coding for adaptive size.

The 4×4 IT has two main advantages. Firstly, it can be implemented with additions and shifts in 16 bit arithmetic only. Secondly, in contrast to floating point arithmetic which gives different results on different platforms, there is no problem of mismatch on the encoder and decoder side for integer arithmetic. A MB is divided into 16 blocks of 4×4 pixels and they are processed one by one.

Transform and quantization processes are combined with each other to save the processing power by avoiding multiplications. Let a 4×4 block is defined as $X = \{x(i,j)|i,j\varepsilon\{0,3\}\}$ as shown in Fig 2.20. First of all, $x(i,j)$ is predicted from its neighboring blocks and we get the residual block:

$$e(i,j) = P(x(i,j), b_1(i,j), b_2(i,j), b_3(i,j), b_4(i,j)), \tag{2.4}$$

where $b_k(i,j)$ are the pixels from reconstructed top and left blocks for *intra* prediction and $P(.)$ is the prediction function. For example, for vertical prediction mode, the prediction will be performed from top block as $P(x,a,b,c,d) = x - a$, where a is the reconstructed block at top.

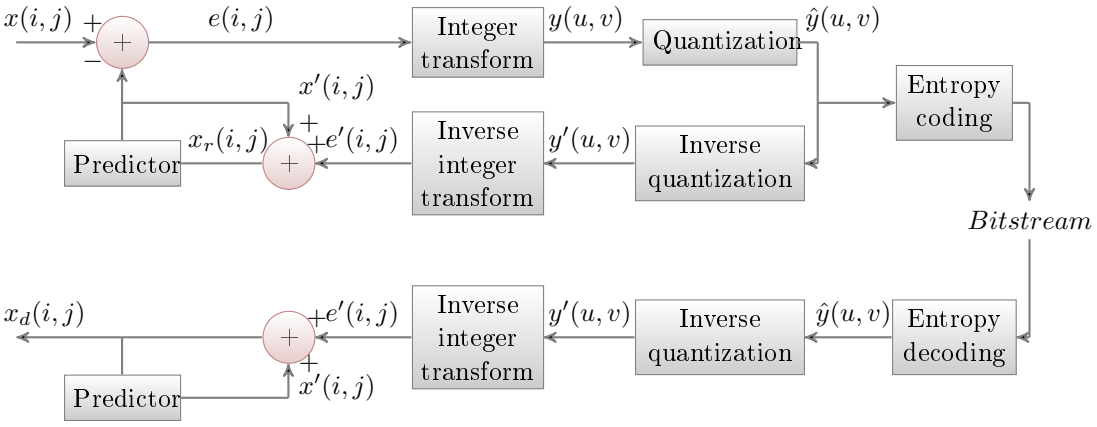


Figure 2.20: Detailed block diagram explaining prediction, transform and quantization steps in H.264/AVC.

Forward and inverse IT 4×4 matrices (A, A_{inv}) are given as [Malvar 2003]:

$$A = \begin{bmatrix} 1 & 1 & 1 & 1/2 \\ 1 & 1/2 & -1 & -1 \\ 1 & -1/2 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \qquad A_{inv} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 1 & -1/2 \end{bmatrix}. \tag{2.5}$$

This residual block E is then transformed using the forward transform matrix A:

$$Y = AEA^T, \tag{2.6}$$

where $E = \{e(i,j)|i,j \in \{0,3\}\}$ is in the spatial domain and $Y = \{y(u,v)|u,v\varepsilon\{0,3\}\}$ is in the frequency domain. Scalar multiplication and quantization are defined as:

$$\hat{y}(u,v) = sign\{y(u,v)\}[(\mid y(u,v) \mid \times Aq(u,v) + Fq(u,v) \times 2^{15+Eq(u,v)})/2^{(15+Eq(u,v))}], \tag{2.7}$$

where $\hat{y}(u,v)$ is a QTC, $Aq(u,v)$ is the value from the 4×4 quantization matrix and $Eq(u,v)$ is the shifting value from the shifting matrix. Both $Aq(u,v)$ and $Eq(u,v)$ are indexed by QP. $Fq(u,v)$ is the rounding factor from the quantization rounding factor matrix. This $\hat{y}(u,v)$ is entropy coded and sent to the decoder side.

On the decoder side, inverse quantization is given by the expression $y'(u,v) = \{[(\hat{y}(u,v)\times(Bq(u,v)\times2^4))\times2^{Eq(u,v)}]+2^3\}/2^4$, where $Bq(u,v)$ and $Eq(u,v)$ are the values from inverse 4×4 quantization matrix and the shifting factor respectively. $y'(u,v)$ is then inverse transformed to get $E' = (A_{inv}Y'A_{inv}^T+2^5)/2^6$. The decoded residual signal $e'(i,j)$ is then added to the predicted signal to reconstruct the original signal back. The detailed explanation of the transform and quantization process of H.264/AVC is well documented in [Richardson 2003].

2.3.3 Context-adaptive variable length coding (CAVLC)

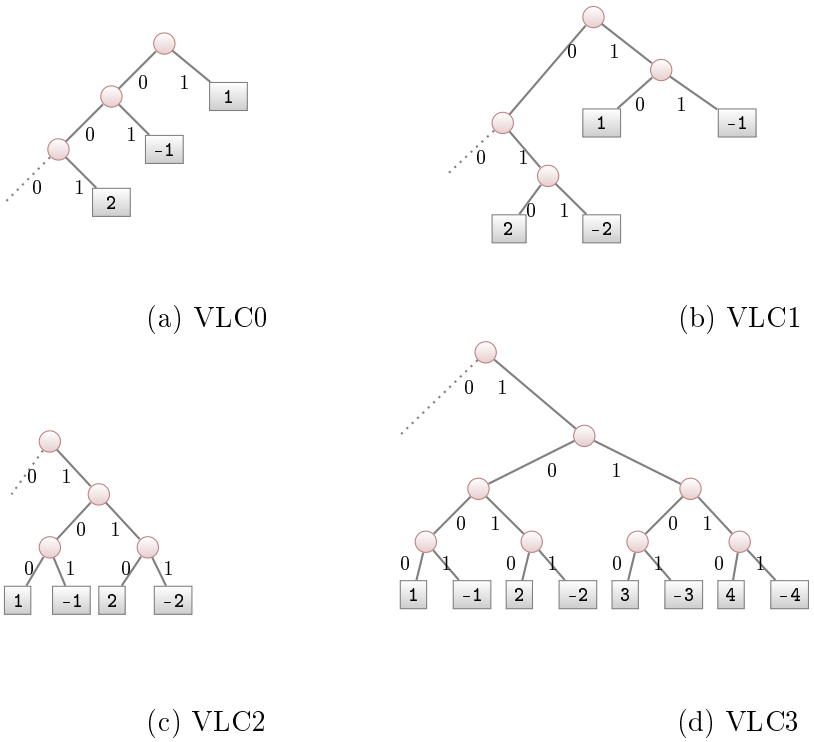
CAVLC is an adaptive entropy coding and is based on Huffman coding. In this entropy coding, run-length coding is performed first as it encodes *levels* and *runs* separately.

To adapt to the local statistical features of DCT coefficients, CAVLC uses seven fixed VLC tables to code *levels* and five fixed VLC tables to code *runs*. The tree representation of first four VLC tables for *levels* is shown in Fig. 2.21. For example, a *level* $'2'$ will be coded as $'010'$ using VLC1 table, while it will be coded as $'1010'$ using VLC3 table. For *level* coding, CAVLC uses two modes: regular and escape. If magnitude of *level* lies within the range of that VLC table, it is coded by regular mode, otherwise escape mode is used. Adaptive nature is introduced by changing the table for the next NZ based on the magnitude of the current NZ as shown in Fig. 2.22. For the first NZ, VLC0 table is used unless there are more than 10 NZs and less than 3 trailing ones(T1's), in which case it is coded with VLC1 table. Similar process is used for coding of *runs*, using different VLC tables and threshold for table transition.

2.3.4 Context-adaptive binary arithmetic coding (CABAC)

CABAC is designed to better exploit the characteristics of NZs as compared to CAVLC, consumes more processing and offers about 10% better compression than CAVLC on average [Moccagatta 2002]. Run-length coding has been replaced by *significant map* coding which specifies the position of NZs in the 4x4 block. Binary arithmetic coding module (BAC) of CABAC uses many context models to encode NZs and context model for a specific NZ depends on recently coded NZs.

CABAC consists of multiple stages as shown in Fig. 2.23.a. First of all, *binarization* is done in which, non-binary syntax elements are converted to binary forms called *binstrings* which are more amenable to compression by BAC. Binary representation for a non-binary syntax element is done in such a way that it is close to minimum redundancy code. In CABAC, there are four basic code trees for *binarization* step, namely the *unary* code, the *truncated unary* code, the k^{th} *order*



(a) VLC0 (b) VLC1

(c) VLC2 (d) VLC3

Figure 2.21: Tree representation of first four VLC tables used in CAVLC. In each VLC table, VLC codes for encircled coefficients have same code length.

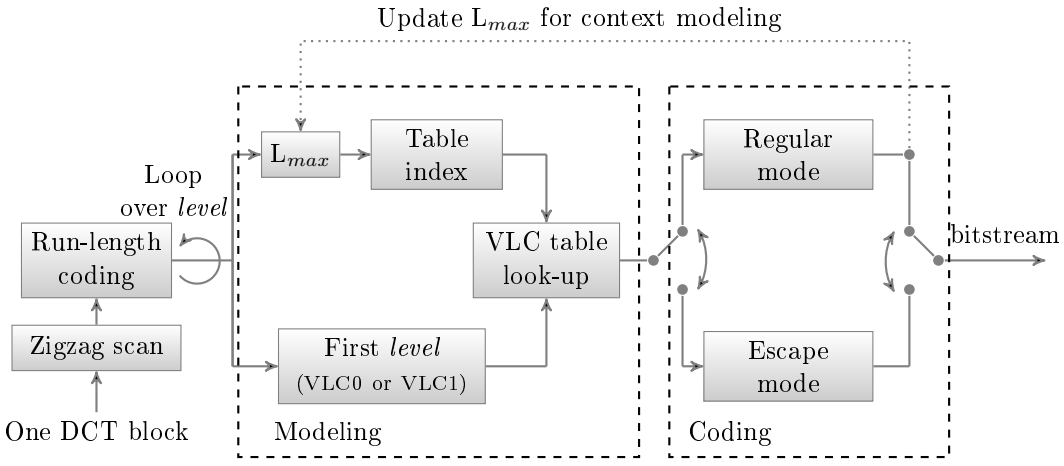


Figure 2.22: Block diagram of *level* coding in CAVLC of H.264/AVC.

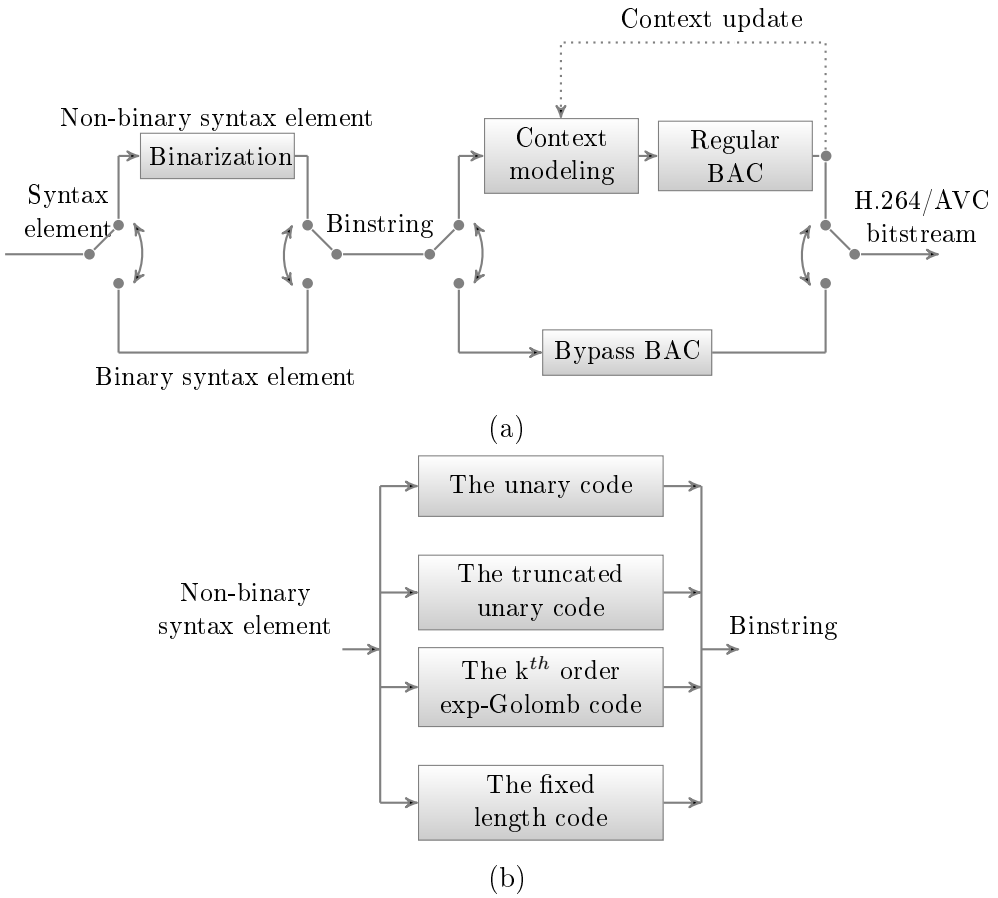


Figure 2.23: (a) Block diagram of CABAC entropy coding module of H.264/AVC, (b) Binarization stage of CABAC.

exp-Golomb code (EGk) and the *fixed length* code as shown in Fig. 2.23.b.

For an unsigned integer value $x \geq 0$, the *unary* code consists of x 1's plus a terminating 0 bit. The *truncated unary* code is only defined for x with $0 \leq x \leq s$. For $x < s$ the code is given by the *unary* code, whereas for $x = s$ the terminating "0" bit is neglected. EGk is constructed by a concatenation of a prefix and a suffix parts and is suitable for binarization of syntax elements that represent prediction residuals. For a given unsigned integer value $x > 0$, the prefix part of the EGk *binstring* consists of a unary code corresponding to the length $l(x) = \left[log_2(\frac{x}{2k} + 1)\right]$. The EGk suffix part is computed as the binary representation of $x + 2^k(1 - 2^{l(x)})$ using $k+l(x)$ significant bits. Consequently, for EGk binarization, the code length is $2l(x)+k+1$. When $k = 0$, $2l(x)+k+1 = 2l(x)+1$. The *fixed length* code is applied to syntax elements with a nearly uniform distribution or to syntax elements, for which each bit in the *fixed length* code *binstring* represents a specific coding decision e.g., *coded block flag*.

In addition to above mentioned binarization techniques, three syntax elements are binarized by concatenation of the basic code trees, namely *coded block pat-*

tern, NZ and the motion vector difference (MVD). Binarization of absolute *level* of NZs is done by concatenation of *truncated unary* code and EG0. The *truncated unary* code constitutes the prefix part with cutoff value $S = 14$. Binarization and subsequent arithmetic coding process is applied to the syntax element $coeff\_abs\_value\_minus1 = abs\_level - 1$, since quantized transformed coefficients with zero magnitude are encoded using *significant map*. For MVD, *binstring* is constructed by concatenation of *truncated unary* and EG3. The *truncated unary* constitutes the prefix part with cutoff value $S = 9$. Suffix part of MVDs contains EG3 of $|MVD| - 9$ for $|MVD| > 9$ and sign bit.

2.4 Audio Video coding Standard (AVS)

AVS [Fan 2004] is the state of the art video coding standard of China. It has slightly less performance but has much less complexity than H.264/AVC [H264 2003]. We are going to present an overview of AVS in Setion 2.4.1, while Setion 2.4.1 presents the entropy coding module of AVS.

2.4.1 Overview of AVS video coding standard

Similar to several other video codecs, AVS is a block based video codec and performs video compression on MBs. Spatial prediction is performed on blocks of 8x8 in *intra* frame, in contrast to 4x4 and 16x16 block size in H.264/AVC. It is less complex with only five modes for *luma* prediction, as compared to thirteen for *luma* prediction modes in H.264/AVC. Reference pixels, which are to be used for prediction, are first low pass filtered in some of the modes of AVS. In *inter* frame, it supports variable block size motion estimation up to 8x8 block, quarter pixel motion estimation and up to two reference frames. DCT has been replaced by Integer Cosine Transform (ICT) [Ma 2006] in AVS. For quantization, a QP value ranges 0-63 with a period of approximate 8.

AVS Part-2 supports several profiles. Two entropy coding modes are supported, namely Context-based 2D variable length coding (C2DVLC) in *Jizhun* profile and context-based binary arithmetic coding (CBAC) in *Jiaqiang* profile [Zhang 2009]. AVS decoder complexity is further reduced by moving the inverse scaling from decoder to encoder module.

In comparison to H.264/AVC, AVS Part-2 *Jizhun* profile has about 3% efficiency loss as compared to H.264/AVC *main* profile in terms of bit saving on HD progressive-scan sequences [Wang 2004a]. 8x8 transform coding, 8x8 spatial prediction, motion compensation up to 8x8 block, 8x8 in-loop deblocking filter and 2D variable length coding are major tools of AVS Part-2 which distinguishes it from H.264/AVC.

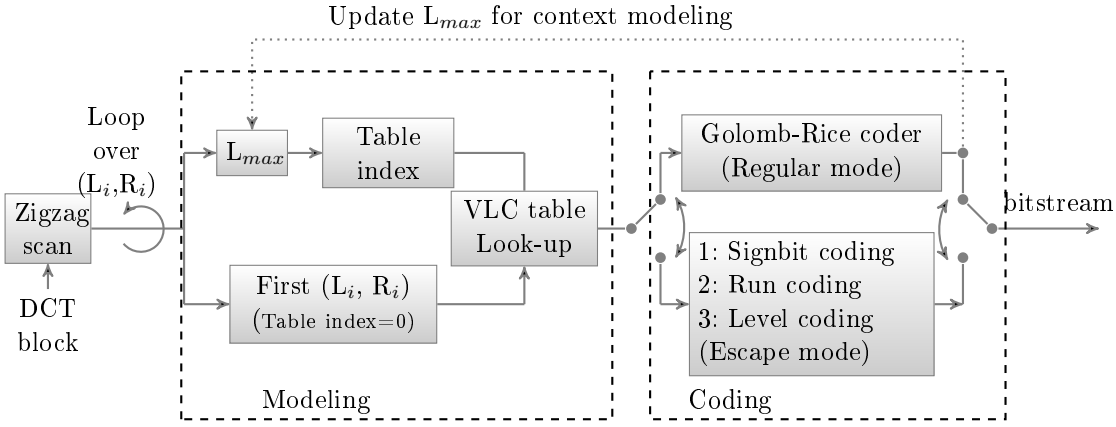


Figure 2.24: Block diagram of C2DVLC entropy coding module of AVS.

2.4.2 Context-based 2D variable length coding (C2DVLC)

C2DVLC is an efficient context-based 2D-VLC entropy coder designed for coding 8x8 block-size transform coefficients. 2D-VLC means that a pair of run-level (L_i, R_i) is regarded as one event and jointly coded [Wang 2006].

Fig. 2.24 illustrates the working of C2DVLC by a flowgraph. A DCT block contains several NZs which are transformed to (L_i, R_i) pairs after the zigzag scan. C2DVLC starts the coding in reverse order using 2D-VLC table with $TableIndex = 0$. Every table has certain range for (L_i, R_i) as shown in Fig. 2.25. If the current (L_i, R_i) lies in the range of current 2D-VLC table, it is encoded by *regular mode*. Otherwise *escape mode* is used.

In *regular mode*, the value of syntax element is firstly mapped to a non-negative integer $CodeNumber$ using a table look-up operation. These $CodeNumbers$ are then mapped to corresponding Exp-Golomb codewords. For (L_i, R_i) pairs having negative value for *level*, $CodeNumber$ is incremented by 1. For example, for $(L_i, R_i) = (2, 1)$ $CodeNumber$ is 11 and for $(L_i, R_i) = (-2, 1)$ $CodeNumber$ is 12. Exp-Golomb codes have regular structures, which means that any non-negative $CodeNumber$ can be mapped to a unique binary codeword using the regular code-constructing rule. Due to the regular codeword structure, the binary code for a given $CodeNumber$ can be constructed in coding process without involving high computational complexity. In AVS, it is a valuable feature that resolves the problem of high memory requirement for multiple VLC tables. In *escape mode*, L_i and R_i are coded separately using Exp-Golomb codewords. R_i and sign of L_i are jointly coded. While for the magnitude of L_i, a prediction is first performed and then the prediction error is coded.

C2DVLC switches the 2D-VLC tables based on the maximum magnitude of the previously coded *levels*. Let L_{max} be the maximum magnitude of the previously coded *levels*. The $TableIndex$ for coding of next (L_i, R_i) is updated if L_{max} is

greater than the threshold of the current table as given below:

$$TableIndex = \quad j, \quad \text{if } (Th[j+1] > L_{max} \geq Th[j]) \tag{2.8}$$

with the threshold for each table given as:

$$Th[0\ldots7] \begin{cases} (0,\ 1,\ 2,\ 3,\ 5,\ 8,\ 11,\ \infty) & intra\_luma \\ (0,\ 1,\ 2,\ 3,\ 4,\ 7,\ 10,\ \infty) & inter\_luma \\ (0,\ 1,\ 2,\ 3,\ 5,\ \infty,\ \infty,\ \infty) & chroma \end{cases} \tag{2.9}$$

This process is repeated for all the (L_i, R_i) pairs. At last, the EOB flag is coded to signal the end of block.

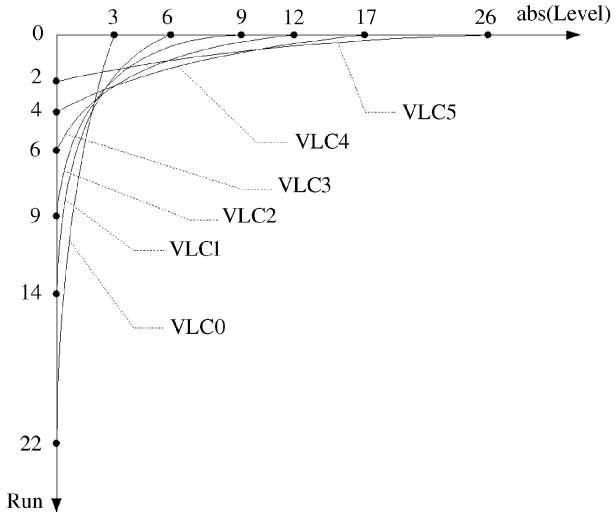


Figure 2.25: Limits of 2D-VLC tables of C2DVLC.

2D-VLC entropy coding has already been used in former video coding standards such as MPEG-2/4. But it has two main differences here. First, Huffman coding has been replaced by Exp-Golomb coding in AVS. Second, former video coding standards are not adaptive and use one single VLC table to code a certain type of transform blocks, e.g. one table for *intra* blocks, one table for *inter* blocks etc.

In AVS, 19 2D-VLC tables have been introduced for coding of residual coefficients and the memory requirement is only about 1k bytes. This method gives gain up to 0.23 dB compared to one-table-for-one-type-of-block coding method [Wang 2004a]. For further details about C2DVLC, please refer to [Zhang 2009].

2.4.3 C2DVLC vs. CAVLC

The common thing between C2DVLC of AVS and CAVLC of H.264/AVC is that both of them are adaptive to the local statistics of DCT coefficients and coding efficiency of both of them is similar. Otherwise, C2DVLC is substantially different as compared to CAVLC. In CAVLC, Exp-Golomb coding is used only for coding of

syntax elements. Transform coefficients are converted to *levels* and *runs*, which are coded **separately** using multiple VLC tables.

While in AVS, Exp-Golomb coding is used for coding all syntax elements including transform coefficients. Transform coefficients are first converted to (L_i, R_i) pairs. These pairs are mapped to *CodeNumber* which is coded using Exp-Golomb codes in *regular mode*. In *escape mode*, L_i and R_i are coded separately using Exp-Golomb codes.

2.5 Scalable video architecture

In scalable video coding, the video bitstream contains a base layer and number of enhancement layers. Enhancement layer are added to the base layer to further enhance the quality of coded video. The improvement can be made by increasing the spatial resolution, video frame-rate or video quality, corresponding to spatial, temporal and quality/SNR scalability. Previous video standards such as MPEG-2 [MPEG2 2000], MPEG-4 [MPEG4 2004] and H.263+ [H263 1998] also contain the scalable profiles but they were not much appreciated because the quality and scalability came at the cost of coding efficiency. SVC based on H.264/AVC has achieved significant improvements both in terms of coding efficiency and scalability as compared to scalable extensions of prior video coding standards. Similar to the previous scalable video coding standards, SVC is also built upon a predictive and layered approach to scalable video coding.

Scalable extension of H.264, also known as scalable video coding (SVC) [Schwarz 2007] is based on pyramid coding architecture. In this kind of architecture, the total spatial resolution of the video processed is the sum of all the spatial layers. Consequently, quality of subsequent layers is dependent on quality of base layer as shown in Fig. 2.26.a. Thus, the process applied to the base layer must be the best possible in order to improve the quality.

Hsiang [Hsiang 2008] has presented a scalable dyadic *intra* frame coding method based on subband/wavelet coding (DWTSB). In this method, LL subband is encoded as the base layer while the high frequency subbands are encoded as subsequent layers as shown in Fig. 2.26.b. With this method, if the LL residual is encoded, then higher layer can be encoded at a better quality than base layer, as illustrated in Fig. 2.26.c. The results presented by Hsiang have proved to be better than H.264 scalable video coding method JSVM [Wiegand 2007] for *intra* frame. In dyadic scalable *intra* frame coding, the image is transformed to wavelet subbands and then the subbands are encoded by traditional H.264/AVC. Since each wavelet subband possesses a certain range of frequencies, zigzag scan is not equally efficient for scanning the transform coefficients in all the subbands.

In spatial scalability, the inter-layer prediction of the enhancement-layer is utilized to remove redundancy across video layers as shown in Fig. 2.27.a. The resolution of the enhancement layer is either equal or greater than the lower layer. Enhancement layer P images can be predicted either from lower layer or from the

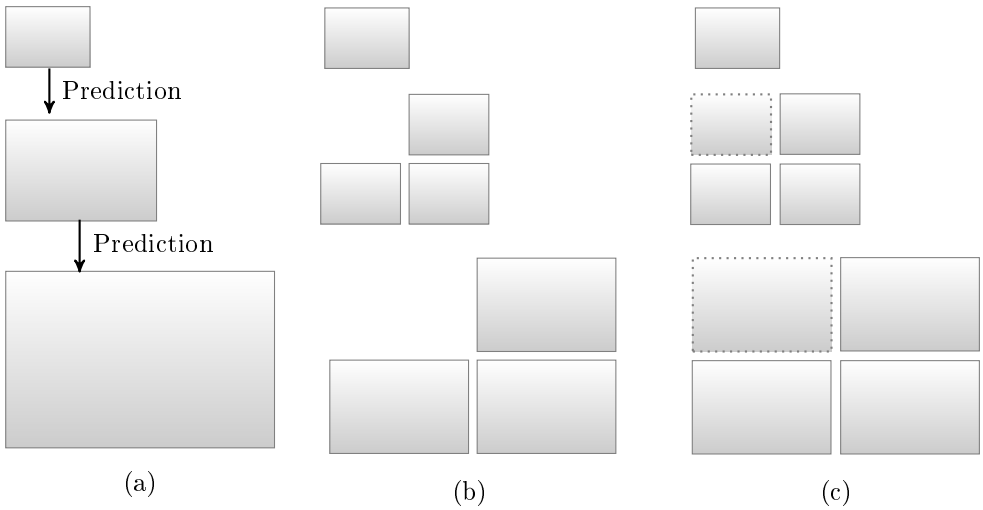


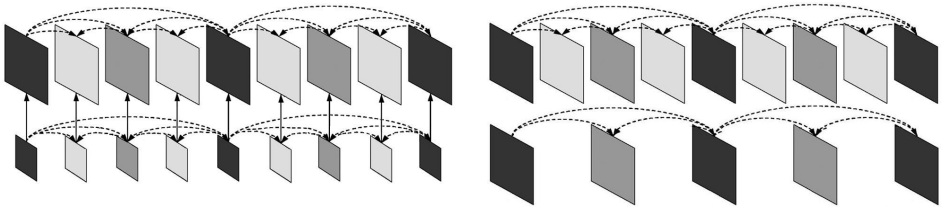
Figure 2.26: Different scalable video coding approaches: (a) Pyramid coding used in JSVM, (b) Wavelet subband coding used in JPEG2000, (c) Wavelet subband coding for dyadic scalable *intra* frame.

previous frame in the same layer. In temporal scalability, the frame rate of enhancement layer is better as compared to the lower layer. This is implemented using I, P and B frame types. In Fig. 2.27.b, I and P frames constitute the base layer. B frames are predicted from I and P frames and constitute the second layer. In quality/SNR scalability, the temporal and spatial resolution of the video remains same and only the quality of the coded video is enhanced.

Applications like digital cinema, motion picture production and satellite imaging requires fast random access to individual video frames. For that purpose, SVC offers a new profile for *intra* frame video coding named 'Scalable High *Intra* Profile'. DWTSB spatially scalable coding framework, presented by Hsiang [Hsiang 2008], for *intra* frames can be used to enhance the efficiency of this profile. This framework is quite flexible in selecting the wavelet coefficients for generating low resolution video at the base layer. In contrast to visual texture coding (VTC) [MPEG4 2004] of MPEG-4 which is based upon separate zero-tree based system for coding wavelet coefficients, DWTSB framework can be integrated to JSVM reference software without much modification. DWTSB based H.264 coding system takes advantage of the benefits of wavelet coding without much increase in implementation complexity.

2.6 Quality metrics

Before presenting various embedding and encryption techniques, it would be expedient to present the metrics to compare quality of visual content before and after embedding. To analyze the quality of the watermarked and encrypted video, with respect to the original, the measure of PSNR (peak signal to noise ratio) has been



(a) (b)

Figure 2.27: Spatial and temporal scalability offered by SVC: (a) Spatial scalability in which, resolution of enhancement layer can be either equal to or greater than resolution of base layer, (b) Temporal scalability in which, first layer containing only I and P frames while second layer contains B frames also. Frame rate of second layer is twice the frame rate of first layer.

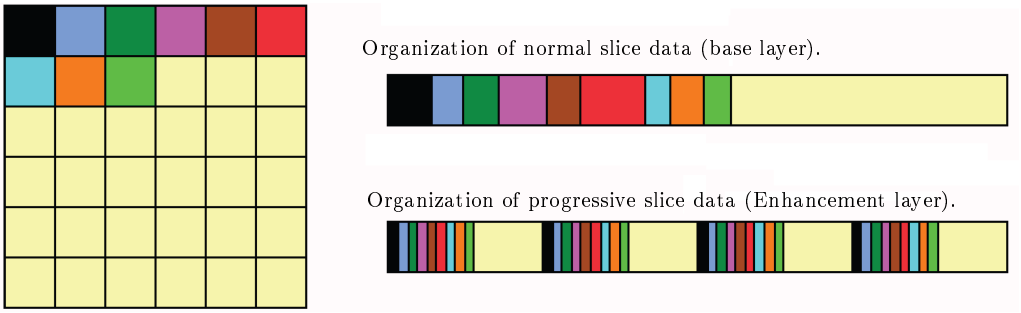


Figure 2.28: SNR scalable architecture of SVC.

employed which is given by:

$$PSNR = 10\ log_{10}\frac{255^2}{MSE}, \tag{2.10}$$

where mean square error (MSE) is a measure used to quantify the difference between the initial video frame I and the distorted video frame I'. If the video frame has a size of $M \times N$ then:

$$MSE = \frac{1}{MN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left(I\left(i,j\right) - I'\left(i,j\right)\right)^2. \tag{2.11}$$

In order to judge compression performance a measure called compression ratio is utilized:

$$Compression\ Ratio = \frac{Original\ Content\ Size}{Compressed\ Content\ Size}. \tag{2.12}$$

As far as error measure is concerned, for a more comprehensive explanation one can rely on the one dimensional version of root mean square error (RMSE) which is the positive square root of MSE: $RMSE = \sqrt{MSE}$.

An interesting measure for assessing the perceptual image quality, to quantify the visibility of errors (differences) between a distorted image (X) and a reference

image (Y), is structural similarity (SSIM) index [Wang 2004b]. It uses a variety of known properties of the HVS and takes into account the similarity with respect to luminance, contrast and structure. The window based SSIM index is given by:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{2.13}$$

where μ_x and σ_x are the pixel mean and standard deviation for a window x from X while μ_y and σ_y represent the same for a window y from Y. σ_{xy} is the covariance between the two windows and the constants C_1 and C_2 are designed to avoid unstable values in case $\mu_x^2+\mu_y^2$ or $\sigma_x^2+\sigma_y^2$ or both approach zero. The overall measure MSSIM (mean SSIM) index, over all the windows to which the images are partitioned, is given by:

$$MSSIM(X,Y) = \frac{1}{W}\sum_{j=1}^{W} SSIM(x_j, y_j) \tag{2.14}$$

Where W is the number of windows.

2.7 Summary

In this chapter, we have given an overview of the evolution of a modern video codec. Starting from the scratch, modern video codec architecture has been developed gradually. Integer transform and entropy coding of H.264/AVC have been presented in detail, since these modules have got developed a lot over the course of time and will be used for watermarking and encryption of video codecs during second part of this manuscript. Important aspects of modern video codec have been uncovered in the above discussion, including the important concept of *intra* prediction in spatial domain, and multiple block size motion estimation to remove spatial and temporal redundancy. After discussion about H.264/AVC, overview of AVS, accompanied by a detailed introduction of C2DVLC, is presented. It is followed by comparison of C2DVLC and CAVLC. The discussion of video codecs is followed by introduction of scalable extension of H.264 (SVC). In the last, we have presented PSNR and SSIM quality metrics, which will be used for analysis purposes in the following chapters of this manuscript. After presenting the architecture of video codecs in this chapter, it is pertinent to present the state of art about watermarking, encryption and fingerprinting of image and video content in next three chapters.

Chapter 3

Image and Video Watermarking

Contents

3.1 Introduction

Avec l'ère de l'informatique, quelques domaines de connaissance qui étaient sur le point de l'extinction, sont de nouveau fortement considérés. Avec le développement récent d'Internet et la disponibilité de puissance de traitement peu coûteuse ainsi que la taille des mémoires, il est très facile de copier, modifier et redistribuer le contenu numérique. Ceci est devenu une préoccupation majeure pour les propriétaires de contenu multimédia. Cela a donné naissance à un nouveau domaine de recherche dans les années 1990, le tatouage. Le tatouage peut-être fragile ou robuste et a plusieurs applications au niveau sécurité et amélioration des média. Au niveau de la sécurité, il peut être utilisé pour l'identification des droits d'auteur, la prise des empreintes digitales active, l'authentification et le contrôle de copie. Dans les applications d'amélioration des média, le tatouage est utilisé pour améliorer

la performance de compression, la détection et correction d'erreur, l'ajout de méta-données et le contrôle d'émission. Ce chapitre vise à présenter les techniques de base, la classification et les progrès récents du tatouage d'image et de vidéo watermarking, avec un intérêt pour le tatouage de vidéos H.264/AVC.

With the Computer age, some areas of knowledge that were on the verge of extinction, are being again highly regarded. With the recent proliferation of the Internet along with the availability of inexpensive processing power and memory size, it is very easy to copy, modify and redistribute digital content. This has become a major concern for multimedia content owners. This gave birth to a new field of watermarking in 1990's. Watermarking can be fragile or robust and has several application for multimedia content regarding security and media enhancement. Regarding security, it can be used for copyright identification, active fingerprinting, authentication and copy control. In application of media enhancement, it is used improve compression performance, improved error detection and correction, meta data and broadcast monitoring. This chapter aims at presenting the basic techniques, classification and the recent progress in the field of image and video watermarking, with a focus on H.264/AVC watermarking.

The rest of the chapter is divided as follows. Section 3.2 lists the fundamental challenges in information hiding followed by a generic classification of the field in Section 3.3. Some embedding techniques are presented in Section 3.4 with special focus on latest embedding techniques. The when and where of data embedding in image/video content with a focus on H.264/AVC is discussed in detail in Section 3.5. This is followed by detailed literature survey in Section 3.6. The concluding remarks are presented in Section 3.7.

3.2 Design challenges

These approaches to protection, whether by encryption or by hiding, are based on known algorithms respecting the Kerckhoffs' principle [Kerckhoffs 1883] and a key shared between the transmitter and receiver. During transit, the data can be attacked, intentionally or otherwise, or subjected to adverse conditions in the case of a wireless transmission, for example. Depending on the application, the choice of technique is determined by the conditions of transmission and reception. The process becomes more and more complex depending on the priority objectives to be taken into account.

The design of a watermarking system involves number of challenges.Bender *et al.* [Bender 1996] have underlined the following restrictions and features during the embedding process:

- Embedding should occur without significant degradation or loss of perceptual quality of the cover.
- For data consistency, original data of the cover rather than header or wrapper

must be used for embedding.

- Intelligent attacks or anticipated manipulations such as filtering and resampling should not mutilate the embedded data.
- If only a part of the cover is available, the embedded data should still be recoverable, i.e. by resynchronization.
- Degradation of the embedded data is always expected as a result of modifications in the cover data. One way to minimize this, may be the employment of error correcting codes.

3.3 Classification

Many watermarking schemes have been proposed in the literature for image and video content. They offer different combination of rate, distortion and robustness. For each application, a watermark algorithm can be selected according to its requirements. For example, applications for copyright protection would require using a robust watermarking, while applications for proving integrity would employ a fragile or semi-fragile watermarking. As shown in Fig. 3.1, watermarking algorithms can be classified as following:

1. *Fragile/Semi-fragile watermarking* provides protection against forgery. Security is the principal issue (*e.g.*, [Li 2003]). Semi-fragile watermarking is used for authenticating of digital material (*e.g.*, [He 2003]).

2. *Robust Watermarking (Copyright marking, traitor tracing, copy control)* protects against the attacks. It must be robust (e.g. [Furon 2000, Wu 2005a, Ramkumar 2000b, Lee 1999]). It can be divided into two main techniques:

   (a) *Non-informed* techniques are those which do not take into account the cover signal statistics *e.g.*, spread spectrum.

   (b) *Informed* watermarking techniques take into account the cover signal statistics.

   On the basis of payload, non-informed and informed watermarking techniques can be classified as:

   (a) *Zero-bit watermarking*: In this type, only watermark detection is performed and there is no message to extract e.g. broken arrows

   (b) *Multi-bit watermarking*: Hidden message is embedded and it is detected and extracted on the decoding side. Well known types of informed multi-bit watermarking techniques are dirty paper trellis codes (DPTC) and quantization index Modulation (QIM).

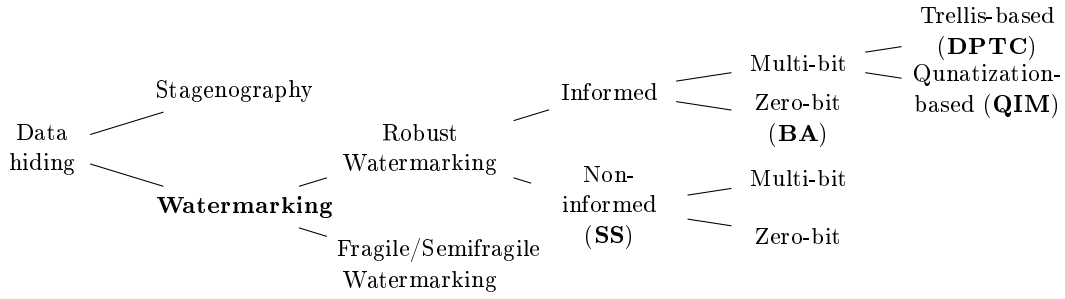


Figure 3.1: General classification of watermarking techniques.

Broadly speaking, all of the above comes down to two, namely the steganography and watermarking. Steganography must be statistically invisible, while no such constraint exists for the latter. In addition, watermarking requires hiding of far lesser payload than steganography and even can have payload of zero-bit. Pixel or coefficient allocation for data embedding may be regular (e.g. every k^{th} pixel) or irregularly distributed (e.g. pseudo-random). Probably the most preferred pixel allocation is by running a pseudo-random number generator (PRNG) using some secret key as a seed. Finally, an embedding method is *blind* if data extraction by the recipient does not require the original cover.

3.4 Embedding techniques

Digital data can be embedded in many ways in the images, e.g. sequential, random, non-random (looking for 'noisy' areas of the image, that will attract less attention), redundant, encrypted, non-encrypted, etc. Each one of these has its own merits and demerits. Based on the manner to embed, one can classify the image watermarking techniques into many classes based on factors like robustness, imperceptibility, choice of embedding areas or domain of embedding [Cox 2008]. On the basis of embedding and detection, the watermarking techniques can be classified as:

- *Non-informed* watermarking techniques do not take into account the host characteristics. Thus the host itself ends up as being the noise, or interference, in the system and payload of such techniques is quite low.

- *Informed* embedding technique takes into account the cover work, since cover work is known during the embedding step. It consists of informed coding and informed embedding as shown in Fig. 3.2. The concept of informed coding is inspired from the work of [Costa 1983], which showed that we can take advantage of availability of cover work to code the message to be embedded. The objective is to attain an optimal trade-off between estimates of perceptual fidelity and robustness. In message coding process, the same message can be mapped to different codes for different cover works.

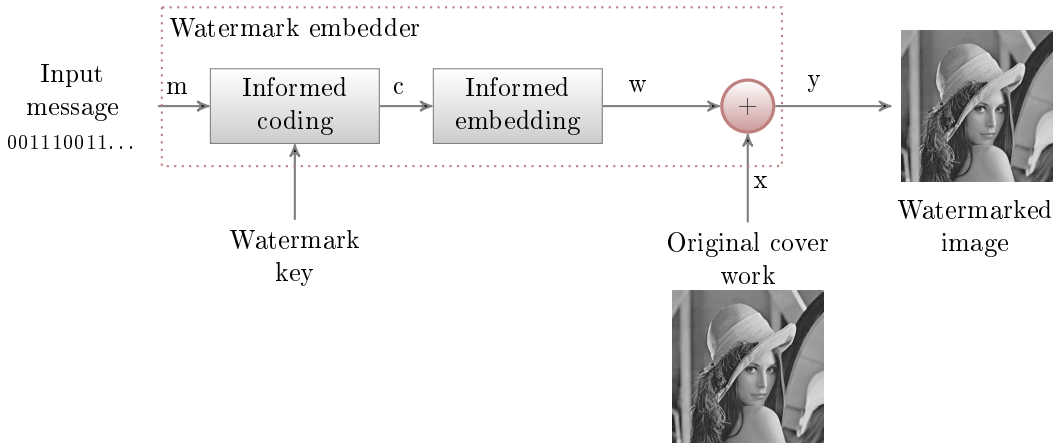


Figure 3.2: Informed embedding.

Here, we are presenting one algorithm from each family for the sake of brevity:

- Least significant bit (LSB) insertion
- Spread spectrum embedding (from non-informed embedding)
- Broken arrows (zero-bit informed embedding)
- Quantization index modulation (quantization based multi-bit informed embedding)
- Dirty paper trellis codes (trellis based multi-bit informed embedding)

3.4.1 Least significant bit (LSB) embedding

A natural way to embed information into a media host without inducing any perceptual distortion is to modify the LSB bit of the media samples. LSB techniques embed the message bits directly into the least-significant bit plane of the cover image in a deterministic sequence [Lin 1999]. Pixel allocation may be sequential or a pseudo-random number generator (PRNG) [Fridrich 2003] may be employed using some secret key.

Probably the most popular, LSB embedding techniques embed data bits in the least significant bits of the image under the assumption that the resultant change would be highly imperceptible due to obvious limitations of human visual system (HVS). A significant amount of information can be embedded without visible loss of quality of the cover image. Fig. 3.3 shows the process of spatial domain LSB embedding when one bit each is embedded in all the LSB's of red green and blue portions of pixel.

The LSB based embedding techniques have many advantages over other techniques *e.g.*, high perceptual transparency and low degradation in the visual quality.

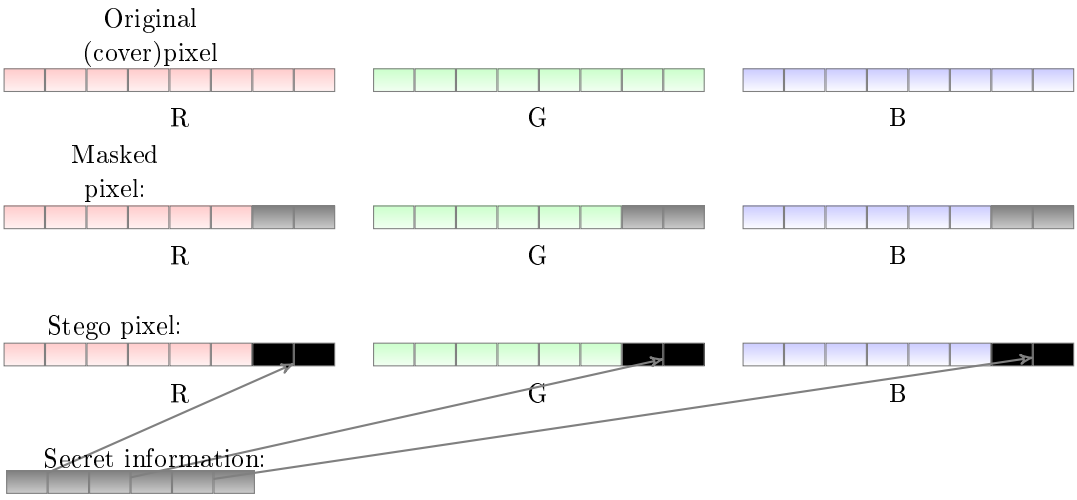


Figure 3.3: An example LSB embedding in the RGB space

But it has also limitations of low robustness to post-processing/malicious attacks, vulnerable to accidental or environmental noise and low tamper resistance.

An example shown in Fig 3.4 illustrates a spatial domain LSB embedding, in a 256×256 gray scale image, the *girl*, with one bit embedded per 16 pixels (1/16 bpp) using pseudorandom allocation. One can see that there is little difference between the original and the embedded image as appear to the naked eye; a fact confirmed by the PSNR which is 63.21 dB.

(a) Original (b) LSB Embedded (c) Difference

Figure 3.4: The *Girl* image subjected to 1 bpp LSB embedding.

3.4.2 Spread spectrum (SS) embedding

Spread-spectrum (SS) hiding is a well known non-informed watermarking technique by Cox *et al.* [Cox 1997]. It was designed to alleviate the problems of LSB hiding against attacks. The method, derived from its communications counterpart, involves adding a spread sequence to the image called a chip [Cox 2008]. The

chip is constructed from the message to be hidden. The method and its variations [Malver 2003, Ruanaidh 1998], proposed for watermarking applications, are robust against many attacks such as compression, noise addition, and signal processing operations. However, in general, it is well-known that the , especially for blind implementations. This is because the additive methods do not utilize the fact that the host is known to the encoder.

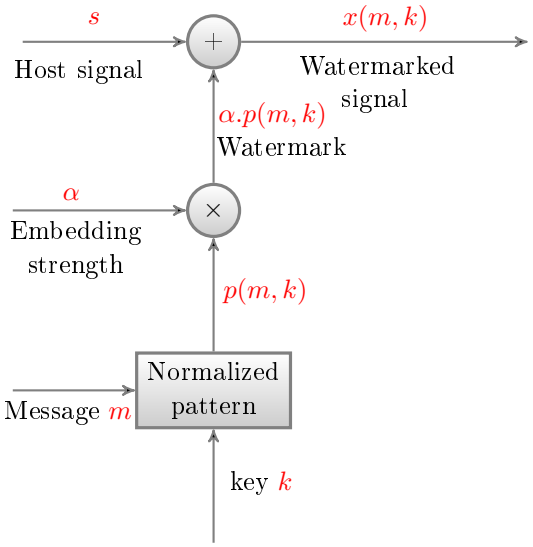


Figure 3.5: General spread spectrum embedding scheme

Fig. 3.5 illustrates a generalized SS encoding procedure. The message m encrypted with a key k, yields a normalized pattern $p(m,k)$. The embedding strength, which is denoted by α, after the mark generation operation yields $\alpha p(m,k)$ that on embedding into the host signal s results in the embedded signal $x(m,k)$:

$$x(m,k) = s + \alpha p(m,k),$$

The decoding process may involve correlation analysis, image restoration techniques and/or error control coding [Marvel 1999]. For decoding, usually a test statistics $t(y,m,k)$ is defined and the aim is to find m that maximize $t(y,m,k)$. Some example statistics are:

1. Likelihood ratio test statistics:

$$t(y,m,k) = \frac{p(y,k|m)}{p(y,k|0)},$$

   which may implemented in case the attack channel is known.

2. Correlation statistics:

$$t(y,m,k) = y.p(m,k),$$

   which is well-suited for blind systems and that is why it is the most popular.

3. Correlation statistics:

$$t(y, m, k) = (y - s).p(m, k),$$

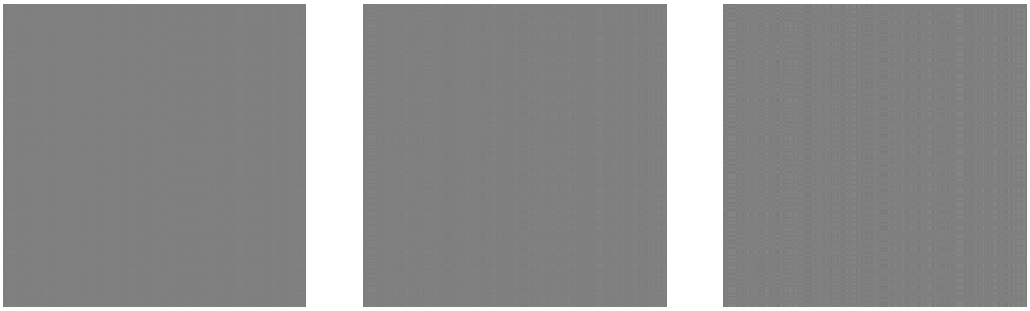
which is suited for nonblind systems.

For better results, α can be made dependent on local properties of s.

As an example take again the image from Fig 3.4.a. and subject to SS embedding at various strengths (α). A message is being embedded in such a way that one bit requires a 8×8 pixel block. As a result one gets the embedded images given in Fig. 3.6 where the deterioration in quality ranges from a PSNR of 48.13 dB to 36.10 dB. This distortion is far greater than the one observed for the LSB embedding example in the Section 3.4.1 where a four times bigger message was embedded. The difference images corresponding to Fig. 3.6 are given in Fig. 3.7.

(a) $\alpha = 1$ (48.13 dB) (b) $\alpha = 2$ (42.11 dB) (c) $\alpha = 4$ (36.10 dB)

Figure 3.6: SS embedding at various strengths in the *Girl* image.



(a) $\alpha = 1$ (b) $\alpha = 2$ (c) $\alpha = 4$

Figure 3.7: Difference brought by SS embedding in the *girl* image at various α.

3.4.3 Broken arrows

Broken arrows (BA) is a robust zero-bit watermarking technique, presented in the international challenge BOWS-2 (break our watermarking system 2nd ed.) [Furon 2008].

The embedding and detection of BA involve four nested spaces: 1) the pixel space, 2) the wavelet subspace, 3) the secret subspace and 4) the MCB (Miller, Cox and Bloom) plane as shown in Fig. 3.8. The main steps for watermark generation are:

- Take the $H_i \times W_i$ matrix i_X of 8-bit luminance values as the original image in the pixel space.
- Perform the 2D wavelet transform (Daubechies 9/7) on three levels of decomposition of i_X. It is followed by selection of coefficients from all the subbands in the wavelet subspace except the LL subband. These $N_s = H_i \times W_i(1-1/64)$ wavelet coefficients are then stored as s_X.
- They use N_v secret binary antipodal carriers signals of size

  $Ns : s_{C,j} \in \{-1/\sqrt{N_s}, 1/\sqrt{N_s}\}^{N_s}, \forall j\{1, \ldots, N_v\},$

  produced by a pseudorandom generator seeded by the secret key K'. The host signal is projected onto these carrier signals: $v_X(t) = s_{C,t}^T s_X$, these Nv correlations being stored as $\mathbf{v}_X(t) = (v_X(1), \ldots, v_X(N_v))^T$. This means that v_X represents the host signal in the secret subspace. We can write this projection with the $N_s \times N_v$ matrix S_C whose columns are the carrier signals:$\mathbf{v}_X = S_C^T s_X$. The norm is conserved because the secret carriers are assumed to constitute a basis of the secret subspace:$||v_X||^2 = s_X^T S_C S_C^T s_X \approx ||s_X||^2$.
- Then the host signal v_X is transferred to the MCB plane. Denote $v_C^* \in R^{N_v}$ as the secret vector in the secret subspace. The basis of the MCB plane is given by (v1,v2) as:

$$\mathbf{v}_1 = \mathbf{v}_C^*, \mathbf{v}_2 = \frac{\mathbf{v}_X - (\mathbf{v}_X^T \mathbf{v}_1)\mathbf{v}_1}{||\mathbf{v}_X - (\mathbf{v}_X^T \mathbf{v}_1)\mathbf{v}_1||} \tag{3.1}$$

  Hence, the MCB plane contains v_C^* and v_X. The coordinates representing the host are $cX = (c_X(1), c_X(2))^T$ with $c_X(1) = v_X^T v_1$ and $c_X(2) = v_X^T v_2$. According to a certain criterion for maximizing the robustness, the watermarked coordinates $cY = (c_Y(1), c_Y(2))^T$ are presented as:

$$c_Y = \begin{cases} (c_X(1) + \sqrt{\rho^2 - c_X(2)^2})^T & \text{for} \quad c_X(2) \leq \rho cos(\theta) \\ c_X + \rho(sin(\theta), -cos(\theta))^T & \text{for} \quad c_X(2) > \rho cos(\theta) \end{cases} \tag{3.2}$$

Here the parameter ρ is related to the embedding distortion constraint, and θ is an angle defining the cone of the detection region. Therefore, the watermark signal in the MCB plane can be represented by $c_W = c_Y - c_X$. In order to go back to the wavelet subspace, c_W is firstly projected in the secret space as $v_W = (v_1, v_2)c_W$. Thereby, the watermarked signal in the secret space is $v_Y = v_X + v_W$. Then, v_W is projected back in the wavelet subspace to get the watermark signal in the wavelet domain s_W, which can be written as $s_W = S_C.v_W$. The watermarking step can then

be written as: $s_Y = s_X + mask.s_W$, where mask denotes the perceptual mask that modulates the watermark signal s_W. In BA, we have: $mask_{BA} = |s_X|$, where $|s_X|$ denotes the absolute value of the wavelet coefficients of the host s_X. This scheme provides perceptually acceptable watermarked pictures.

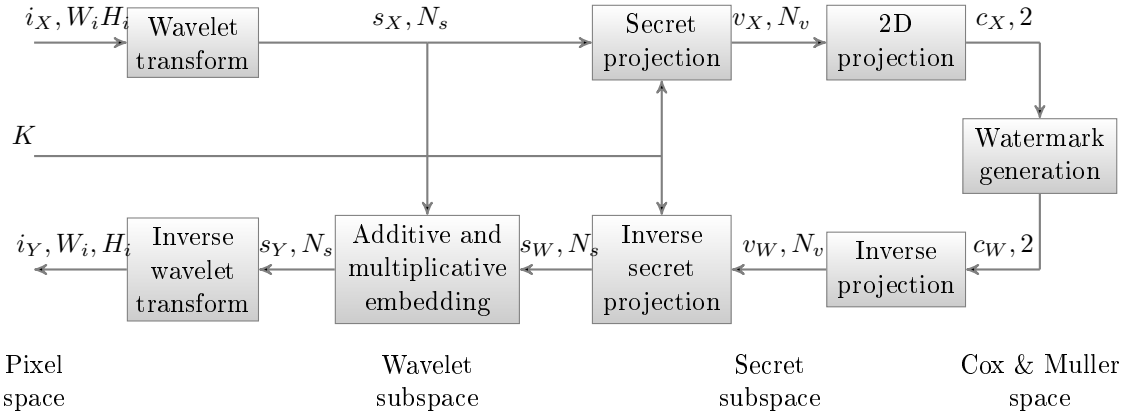


Figure 3.8: Block diagram of broken arrows scheme, showing different subspaces

3.4.4 Dirty paper trellis coding (DPTC)

Dirty paper coding is the most common informed watermarking technique. In this technique, the initial set of codewords is divided into different cosets, which contain several, dissimilar codewords representing the same single message. With such a code, for a given message, the embedder searches through the coset associated with this message and outputs the message mark which fits the best the Original Work e.g. in terms of correlation score. This optimal mark is then embedded. On the receiver side, the detector tests all the possible message marks and extracts the one which fits the best the Work. It then identifies the hidden message by looking to which coset this mark belongs to. Theoretical analysis has shown that, under certain simplifying assumptions, this type of system should be able to obtain the same performances as systems that use informed detection. In other terms, the Original Work, which is often considered as noise in the watermarking perspective, should have no effect on the ability of the detector to correctly extract the watermark.

There are several practical approaches that have been proposed to implement dirty paper codes. Lattice codes are simple to implement, but they are inherently weak against volumetric scaling. That motivates further research in order to obtain new practical dirty paper codes. The proposed approach is based on convolutional codes. A trellis-based coding scheme for dirty paper code is the most efficient technique. Traditionally, in the trellis representation of such a code, there are two arcs exiting from each state as shown in Fig. 3.9. One regular arc has a label which ends with a 0 and one bold arc has a label which ends with a 1. On the other

hand, each state can be reached by two arcs: one bold arc and one regular arc. The number of states in the trellis is then defined by the cross-symbol interference. The more the symbols interfere one with another, the more states in the trellis are required in order to reliably encode the message.

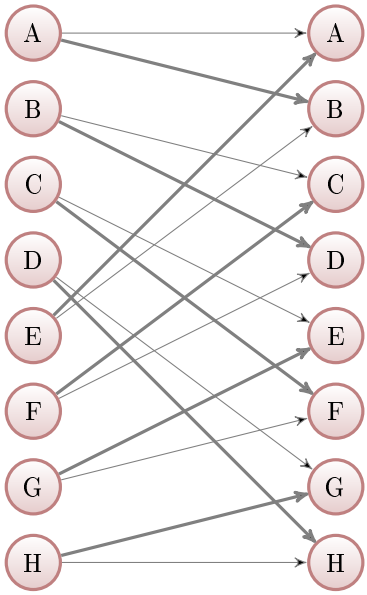


Figure 3.9: Trellis representation of a traditional eight-state convolutional code. The bold arcs have a label 1, while the regular ones have a label 0.

3.4.5 Quantization index modulation (QIM) embedding

Looking at information hiding in multimedia hosts as communication with encoder side information, new embedding methods have been proposed that reduce the host signal interference. These methods are also based on Costa's work on writing on dirty paper [Costa 1983]. In Costa's setting, there is a Gaussian side information known only to the encoder, but not to the decoder (i.e., a paper with Gaussian dirt). There is a Gaussian noise which gets added to the paper before it reaches the decoder. Costa showed that there is no loss of embedding rate thanks to the presence of encoder side information, if codewords are well chosen and the embedding is well achieved. Based on this work, a new class of embedding schemes, called quantization index modulation (QIM), was proposed by Chen and Wornell [Chen 1999, Chen 2001]. The data is hidden by the choice of quantizer (based on the message to be hidden) at the encoder. The decoder just determines which of the possible quantizers were used.

While the methods proposed by Chen and Wornell are based on vector quantizers, simplified version of the schemes employing scalar quantizers have been proposed and applied to multimedia hosts [Solanki 2004, Eggers 2003]. In [Solanki 2004], it is shown that there is roughly only a 2 dB penalty in terms of resilience to attacks

for using scalar quantization as compared to vector quantization. When there are only two possible symbols in the alphabet, the method reduces to the well known odd-even embedding, where the odd reconstruction points represent, say a 1 being embedded, and even reconstruction points represent a 0.

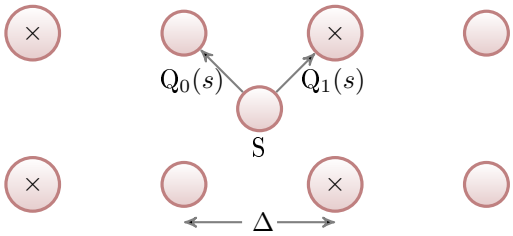


Figure 3.10: Quantization index modulation technique. 1-bit message $m \in [0, 1]$ is inserted using two quantizers Q_0 and Q_1. The step size is Δ.

It should be noted that any of the above techniques, LSB, SS, or QIM, can be employed on the pixels of the image (i.e., the spatial domain), or the transform coefficients. For LSB embedding in transform domain, such as DCT or DWT, the coefficients must have been already quantized (i.e., compressed).

3.4.6 Psycho visual masking

For multimedia watermarking, it is necessary to be adaptive to the local characteristics of the host signal, since all the parts of the host image do not have the same hiding capacity [Cox 2008]. In other words, we cannot make same amount of change in all the portions of an image to hide data. Thus, any scheme embedding data into a media host must provide a way to adapt to local perceptual characteristics. Commonly used perceptual models are of Watson's DCT and wavelet models [Watson 1993, Watson 1997] that determine the strength of the watermark based on its perceptual sensitivity. The use of perceptual shaping for image-adaptation remains the most popular approach to maintain perceptual transparency and many methods for perceptual adaptation have been proposed in the literature. Early works which use perceptual masking include [Chae 1999, Wolfgang 1999, Podilchuk 1998]. For DCT based watermarking, now it is well accepted that hiding in DCT coefficients whose values are zero (after quantization), should not be used for embedding information.

For SS embedding, perceptual masking has been used by several researchers to increase its imperceptibility [Wolfgang 1999, Podilchuk 1998]. The strength of the watermark can be controlled by scaling factor α.

One of the earlier work on adaptive quantization based embedding was by Ramkumar [Ramkumar 2000a], in which the zero-valued DCT coefficients were not modified. However, this method was designed for JPEG compression attack only, and it could not survive any other attacks. Also notable is the work by Mukherjee*et al.* [Mukherjee 2000] who use lattice quantization to embed data and pro-

vide image adaptation by choosing different lattice structures for different types of blocks. A perceptual model determines the level of embedding in a block. Wu and Lui [Wu 2003a, Wu 2003b] propose an adaptive method for QIM, called uneven embedding, in which the encoder chooses the hiding locations based on a perceptual model. In their implementation, either the information about the embedding locations is sent as side information (variable embedding rate), or the rate is fixed and embedding locations vary by shuffling (constant embedding rate). They can embed 1024 bits in a 512x512 image that survives JPEG compression attacks and moderate additive noise attacks.

Solanki *et al.* [Solanki 2004] have presented an image-adaptive scheme for watermarking. Their scheme has high-volume watermarking capacity and uses turbo-like codes for erasure and error correction. It offers payload of several thousand bits and is robust against image compression, image tampering, and additive noise.

3.5 The when and where of embedding in H.264/AVC

Being an active research area for the last two decades, watermarking is now an established field and that is why a lot has been written about it [Cox 2008]. We, therefore, focus on the literature about DCT-based watermarking which is again very extensive and one is compelled to be brief and limit oneself to video codecs, as far as possible. Looking at the structure of H.264/AVC codec, as explained in Chapter 2 makes one think about when and where to interrupt the coding flow in order to embed the message. Theoretically, you can interrupt the codec anywhere for embedding but, at the periphery, the embedding capacity is lower, accompanied by relatively higher distortion. There is normally five types of working domains as shown in Fig 3.11. Every type of intervention has its advantages and limitations:

1. *Pre-compression stage (before video encoder)*: In first class of watermarking, message embedding is performed in the spatial domain [Cox 1997, Li 2007]. Spatial domain watermarking techniques are simple and fast to implement but normally are not robust against filtering and lossy compression. Here the watermarking is separated from compression process. So if the embedding technique is robust (e.g. SS in spatial domain) and can withstand compression, it can also withstand transcoding. In this domain, watermarking can be performed on a single video frame or sequence of video frames. Some researchers have embedded the hidden message in the frequency domain like DFT [Kang 2003, Deguillaume 1999] or DWT [Wu 2000].

2. *Codec structure domain*: In this class, hidden message is embedded in the video codec structure (e.g. [Dai 2003, Linnartz 1998]). Watermarking techniques that utilize video codec structure are primarily motivated by the goal of integrating watermarking and compression to reduce overall real-time video processing complexity.

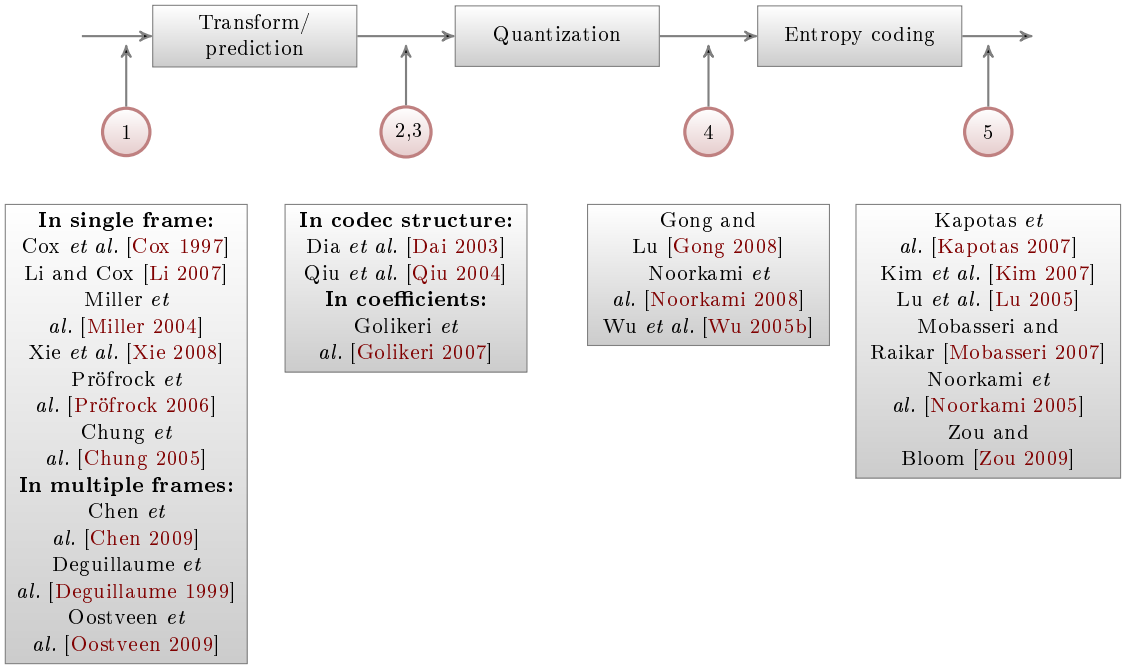


Figure 3.11: Classification of watermarking schemes on the basis of working domain: (1) Spatial, (2) Video codec structure, (3) Transform domain, (4) Quantized transform domain, (5) Bitstream.

3. *Transform domain*: Some researchers have investigated the performance of watermark insertion in DCT coefficients before quantization. For example, it has been proposed to embed in H.264/AVC before quantization step in [Golikeri 2007].

4. *Quantized transform domain*: In this domain, hidden message embedding is performed in quantized DCT coefficients (QTCs).

5. *Bitstream domain*: In this domain, message is embedded directly into compressed bitstream [Arena 2000, Hartung 1998]. In [Cross 2002], Cross and Mobasseri proposed to embed message in entropy coding stage. The main limitation of this technique is that payload is very low and these techniques are not robust to re-encoding with different parameters.

3.6 Prior work of image/video watermarking methods

Watermarking along with compression is primarily motivated by the goal of integrating watermarking and compression to reduce overall real-time video processing complexity. Researchers have presented schemes to perform watermarking in all the candidates domains as explained in Section 3.5. In this section, literature review is

presented separately for each embedding domain.

3.6.1 Embedding in the pre-compression domain

In pre-compression domain watermarking, message embedding is performed before the compression process [Deguillaume 1999, Ejima 2000, Cox 1997, Kang 2003, Deguillaume 1999]. It is marked as stage 1 in Fig. 3.11. Embedding can be performed either in pixel domain or in some transform domain *e.g.*, DCT, DFT, DWT. Temporal aspect of video can also be exploited for watermark embedding by taking into account multiple frames at the same time. In spatial domain, LSB modification is a very simple method to embed hidden message into the cover object [Katzenbeisser 2000]. This method may survive against attacks such as cropping but any addition of noise or lossy compression is likely to defeat the message extraction. In [Katzenbeisser 2000], a method has proposed to use PRNG to decide the pixels for LSB substitution. It will improve the security of the watermarking but it will be still vulnerable to replacing the LSB's with a constant value. LSB modification is perceptually indiscernible but statistically it is discernible. In [Chandramouli 2001], an upper bound for the LSB payload has been defined so that it remains statistically invisible.

In [Chung 2005], transform domain has been used for LSB embedding . Here the signature image was first quantized using vector quantization in order to hide larger message. Binary bits of message were embedded directly into LSB of DCT coefficients in an image. Using the duplication of copies of the message provides further robustness against the signal processing attacks.

In [Pröfrock 2006], Pröfrock *et al.* have presented a watermarking technique in the spatial domain, which is robust to H.264/AVC video compression. Hidden message is embedded in the perceptually significant parts of the video in an imperceptible manner, by changing the spatial position of object borders. Borders are defined by new normed center of gravity (NCG). Influence of lossy compression on NCGs is predicted and watermark is embedded with a defined robustness to lossy compression. A geometric warping process is proposed to quantize the NCG and embed the watermark payload with a defined robustness. Xie *et al.* [Xie 2008] have also proposed robust watermarking based on on-off keying scheme which uses DWT transform to embed the transform.

3.6.2 Embedding in the video codec structure

Video codec structure is the second candidate domain for watermarking. Some researchers have proposed to embed the message in motion vectors [Dai 2003, Qiu 2004]. In [Li 2009], Li *et al.* propose to perform robust watermarking for H.264/AVC by embedding the hidden message in a new syntax element named reference index. They also modify the current block to improve the robustness of the scheme by geometry method with the least degradation in the video quality. These video watermarking techniques are vulnerable to re-encoding and conversion

to other video codecs.

3.6.3 Embedding in the transform domain

Hidden message can also be embedded in transform coefficients before quantization as proposed by Golikeri *et al.* [Golikeri 2007]. It is shown by stage 3 in Fig. 3.11. They have proposed robust watermarking algorithm before quantization for H.264/AVC. They have used visual models developed by Watson [Watson 1993] to choose the coefficients to be watermarked based on frequency sensitivity, luminance masking and contrast masking. They embed message in transformed coefficients before quantization.

3.6.4 Embedding in quantized transform domain

Some researchers have proposed algorithms to embed hidden message in QTCs of H.264/AVC, as shown by stage 4 in Fig. 3.11. For example, Noorkami and Merserau have presented a technique to embed message in both *intra* and *inter* frames in all non-zero QTCs [Noorkami 2008]. They claim that visual quality of *inter* frames is not compromised even if we embed message in all non-zero QTCs. Owing to embedding of message only in non-zero QTCs, their method does not affect the compression efficiency of run-length coding, but performance of context-based adaptive variable length coding (CAVLC) gets affected and as a result, a controlled increase in bitrate is observed, since there are lot of QTCs whose magnitude is 1 and CAVLC encodes *trailing ones* (T1's) separately. In [Gong 2008], Gong and Lu embedded watermarks in H.264/AVC video by modifying the quantized DC coefficients in *luma* residual blocks. To increase the robustness while maintaining the perceptual quality of the video, a texture-masking-based perceptual model was used to adaptively choose the watermark strength for each block. To eliminate the effects of drift, a drift compensation algorithm was proposed which add the drift compensation signal before embedding the watermark bit.

3.6.5 Embedding in the bitstream

To avoid processing intensive decoding followed by re-encoding along with watermarking, some methods have suggested embedding message in compressed bitstream [Noorkami 2005, Lu 2005, Kim 2007, Mobasseri 2007, Zou 2009]. Kim *et al.* suggested to hide the message in the sign bit of the Trailing Ones in CAVLC of H.264/AVC [Kim 2007]. Bitrate of the watermarked video remains exactly the same with PSNR greater than 43 dB. In [Mobasseri 2007], authentication of H.264/AVC is performed by direct watermarking of CAVLC codes. Zou and Bloom [Zou 2009] have proposed to perform direct replacement of CAVLC for watermarking purpose. Kapotas *et al.* [Kapotas 2007] have presented a watermarking method in H.264 streams for fragile watermarking. It takes advantage of the different block sizes used by the H.264 encoder during the *inter* prediction stage in order to hide the desirable data. The message can be extracted directly from the encoded stream

without the need of the original host video. This approach can be mainly used for content-based authentication. Such algorithms face two major limitations. First, payload of such algorithms is very less which is up to few bytes per second as explained in [Hartung 1998]. Second, there is a continuous drift which degrades the visual quality significantly.

3.7 Summary

We discussed, in this chapter, the when and where of watermarking. Before presenting the state of the art, we pointed out the three important factors of payload, robustness and imperceptibility. Keeping in view the main classification of information hiding, we discussed in some detail one representative embedding technique for each, i.e LSB, SS, BA, QIM, DPTC. It is followed by explanation of psycho-visual masks for watermarking. But the main thrust of this chapter was on image and video watermarking with a focus on H.264/AVC in the latter part of the chapter.

Chapter 4

Image and Video Encryption

Contents

4.1 Introduction

Avec l'augmentation du nombre de caméras de surveillance pour la sécurité, la vie privée est devenue une préoccupation importante. Selon une évaluation, il y a plus de quatre millions de caméras de surveillance seulement dans la ville de Londres et un citoyen commun est capturé par 400 caméras en moyenne, pendant une journée normale, par exemple de la maison au bureau. Une demande importante exigée est de limiter l'accès seulement aux personnes autorisées. Ces problèmes sont liés aux organismes chargés de faire respecter la loi. Dans l'industrie du spectacle, il y a une perte économique énorme à cause de la distribution illégale de vidéos. Ces exemples viennent de secteurs différents : d'abord au niveau de sécurité des vidéos à bas débit, tandis que la seconde du monde du spectacle exige de vidéos à haut débit. Mais ces deux exemples ont besoin du même niveau de service concernant la confidentialité et l'accès au contrôle de données visuelles. La raison de la restriction est la protection de la vie privée pour le premier exemple, alors que la protection

commerciale concerne le second exemple. Dans de tels scénarios, le cryptage des données peut être utilisé pour limiter l'accès des contenus multimédia seulement aux utilisateurs autorisés. Dans ce chapitre, nous présentons les techniques de base et l'état de l'art dans le chiffrage d'images et de vidéos.

With the inundation of surveillance cameras to the security prone modern world, the privacy has become an important issue. According to one estimate, there are more than four million surveillance cameras only in London metropolitan and a common citizen is captured by 400 cameras on average, during his normal daily routine e.g. from home to office. It is hence a required feature to restrict the access only to authorized persons. This story is related to law-enforcement community. In the entertainment industry, there is a huge economic loss because of illegal distribution. Both of these examples come from different areas: first from low bitrate security regime, while second from high bitrate entertainment industry. But they need the same feature, i.e. confidentiality and access control of visual data. The reason for this restriction is protection of privacy in first example, whilst protection of the commercial money in the second. In such scenarios, data encryption can be used to restrict the access of multimedia content only to authorized users. In this chapter, we introduce the basic techniques and state of the art in image and video encryption.

This rest of the chapter is arranged as follows. Section 4.2 lists the fundamental challenges in encryption along with a generic classification of the field. It is followed by introduction to block ciphers in Section 4.3, with a focus on Advanced Encryption Standard (AES) and working modes of block ciphers. The context level classification of selective and partial encryption of image/video content with a focus on H.264/AVC is presented in Section 4.4. Section 4.5 then gives detailed literature survey. The concluding remarks have been presented in Section 4.6.

4.2 Classification of encryption algorithms

Cryptography has a significant historical influence since the time of the Pharaohs. In *The Kama Sutra* of Vatsyayana, it is recommended that women should study the art of understanding writing in cipher. Traditionally cryptography has been used by governments and military. It was used extensively in World War I. The decoding by British naval intelligence of the *Zimmermann telegram* helped to bring the United States into the war. Trench codes were used by most of the field armies (Americans, British, French, German and others) in World War I. While, in World War II, many cipher systems were used by different nations. A comprehensive reference of the art of cryptography is presented in *The Codebreakers* [Kahn 1967] by David Kahn in 1967.

The main classical cipher types are transposition and substitution ciphers. Transposition ciphers rearrange the order of letters in a message *e.g.*, *'vous allez bien?'* becomes *'uovs lalez ineb?'* in a trivially simple rearrangement scheme. Sub-

stitution ciphers, on the other hand, systematically replace letters or groups of letters with other letters or groups of letters *e.g.*, *'ça va bien'* becomes *'db wb cjfo'* by replacing each letter with the one following it in the Latin alphabet. Caesar cipher was one of the early substitution ciphers, in which each letter was replaced by a letter some fixed number of positions further down the alphabet.

After World War II a completely new era of communication began. With the introduction of computers and digital messages, enormously enhanced computational power brought by computers broke almost all classical ciphers. This is where classical cryptography ends and modern cryptography begins. Nowadays, cryptography is not only used by military and government organizations, but also by private sector.

Modern cryptography is the conversion of data into a secret code. It enables the sender to securely store sensitive information or transmit it across insecure networks. Hence it cannot be read by anyone except the intended recipient. The original data is called plaintext and the encrypted data is called ciphertext.

According to Kerckhoff's principle, security of encrypted data is entirely dependent on two things: the strength of the cryptographic algorithm and the secrecy of the key [Kerckhoffs 1883]. The key is used for encryption and decryption and must be kept secret, thereby requiring the sender and receiver to agree on the same key before making any data transmissions. While cryptography is the science of securing data, cryptanalysis is the science of analyzing and breaking secure data, and therefore it is the process of recovering the plaintext or key, usually by using the ciphertext and knowledge of the algorithm. Security of a cryptographic algorithm is measured in terms of time and resources it would require to recover the plaintext.

A single cipher, which is ideally suited for all applications, does not exist. Even a cipher offering a high level of security is not the best choice in all practical situations. This is a result of inevitable trade offs required in practical applications, including those arising from, for example, speed requirements and memory limitations (*e.g.*, code size, data size, cache memory), constraints imposed by implementation platforms (*e.g.*, hardware, software, chip cards), and differing tolerances of applications to properties of various modes of operation. In addition, efficiency must typically be traded off against security. Thus, it is beneficial to consider a number of candidate ciphers for a specific application.

Data encryption techniques can be classified as symmetric or asymmetric. Asymmetric data encryption is also known as public key cryptography, while symmetric data encryption is also known as private key cryptography. In Section 4.2.1, a discussion about asymmetric cryptography is outlined, while an overview of public key cryptography is presented in Section 4.2.2.

4.2.1 Symmetric cryptography

Symmetric or private key cryptography is the most widely used encryption technique. In symmetric key cryptography, encryption and decryption are performed using the same secret key as shown in Fig. 4.1. The key must only be known by the sender and receiver to maintain integrity. The primary disadvantage of symmetric

key algorithms is that the key must remain secret at all times. For this reason, the key must be protected and secured requiring the sender to transmit the key to the recipient in a secure fashion [Whitman 2007]. Public key cryptography can be used to exchange the secret key between the parties in a private key cryptography.

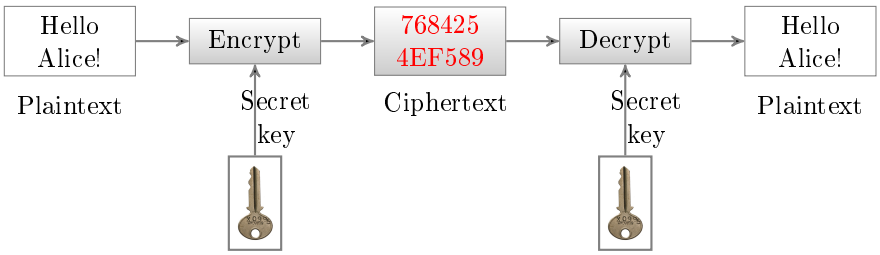


Figure 4.1: Block diagram of private key cryptography.

4.2.2 Asymmetric cryptography

Asymmetric or public key cryptography uses one key for encryption and another for decryption as illustrated in Fig. 4.2. The encryption key (also known as public key) is known to all parties, while the decryption key (also known as private key) is secret. Each user creates a pair of keys: one for encryption, while the other for decryption. The public and private keys are mathematically related and data encrypted with the public key can only be decrypted with the relevant private key. This guarantees message privacy in encrypted form. In public key cryptography, trap-door one-way functions are used. A trapdoor one-way function is a one-way function for which the inverse direction is difficult without a certain piece of information (i.e. the trapdoor, in our case it is the key).

In public key cryptography, public key is known while private key is kept secret by the receiver. The sender uses the recipient's public key to encrypt a message. The recipient uses the private key to decrypt the received message.

Typical examples of asymmetric ciphers are RSA, ElGamal and elliptical curves ciphers [Schneier 1995]. Long keys are used for making the system more secure, because a part of the key (*i.e.* public key) is already known. Typical key lengths are 768, 1024, 2048 bits (RSA, ElGamal). Keys based on elliptical curves have typical lengths of 150 to 200 bits. Key lengths of different algorithms cannot be compared directly because they rely on different mathematical properties, e.g. some researchers say that 170 bits for elliptical curves provide roughly the same security as 1024 bits for RSA.

4.3 Block ciphers

Block ciphers encrypt the plaintext block by block. This is in contrast to stream ciphers which encrypt bit by bit or byte by byte. Symmetric-key block ciphers are

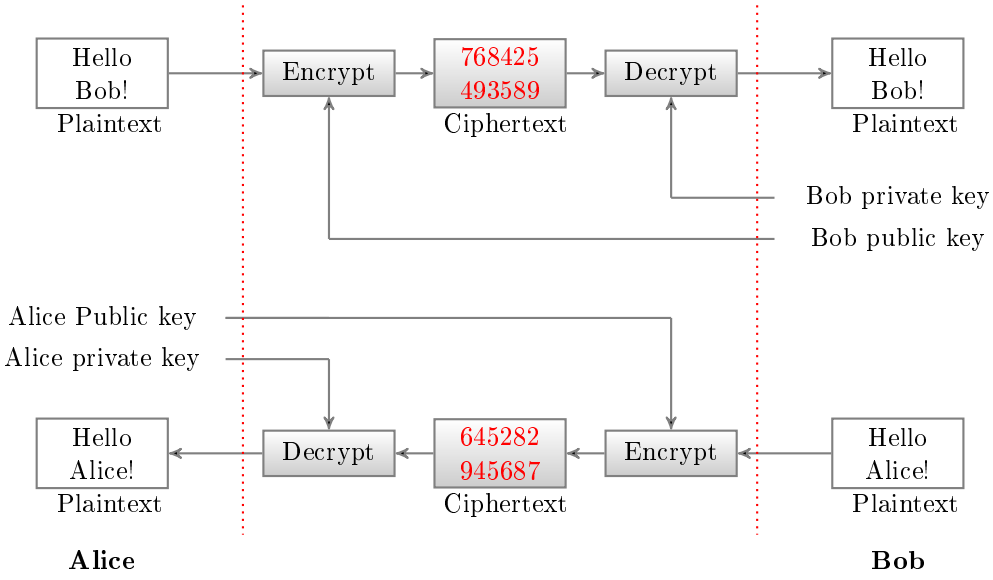


Figure 4.2: Working of public key cryptography.

the most prominent and important elements in many cryptographic systems. In addition to their basic application of confidentiality, they are also used as a fundamental building block for construction of pseudorandom number generators (PRNG), stream ciphers and hash functions. They may furthermore serve as a central component in message authentication codes (MACs), data integrity mechanisms, entity authentication protocols, and (symmetric-key) digital signature schemes.

Among the well known block ciphers, AES [Daemen 2002] is state of the art. Before AES, Data Encryption Standard (DES) [T1.801.03-2003 2003] and 3DES have been in use. FEAL (the Fast data Encipherment ALgorithm) [Shimizu 1988] has received both serious commercial backing and a large amount of independent cryptographic analysis. IDEA (International Data Encryption Algorithm) [Lai 1992] was originally proposed as a DES replacement and has been widely known and highly regarded.

Of the many block ciphers currently available, focus is given to AES in this section since it provides the highest security. We review the basics steps of the AES algorithm in Section 4.3.1, while in Section 4.3.2 we present the working modes of block cipher with an emphasis on cipher feedback (CFB) mode.

4.3.1 The AES encryption algorithm

The Advanced Encryption Standard (AES) algorithm consists of a set of processing steps repeated for a number of iterations called rounds [Daemen 2002]. The number of rounds depends on the size of the key and the size of the data block. The number of rounds is 9 for example, if both the block and the key are 128 bits long. The same number is 11, if either the block or the key is 192 bits long or if neither of

them is longer than that. If either the block or the key is 256 bits long the number of rounds is 13. Given a sequence $\{X_1, ..., X_n\}$ of bit plaintext blocks, each X_i is encrypted with the same secret key k producing the ciphertext blocks $\{Y_1, ..., Y_n\}$ as described in Fig. 4.3.

To encipher a data block X_i in AES, an AddRoundKey step is performed first by XORing a subkey with the block. The incoming data and the key are added together in the first AddRoundKey step. Afterwards, it follows the round operation. Each regular round operation involves four steps which are SubBytes, ShiftRows, MixColumns and AddRoundKey. In the SubBytes step, each byte of the block is replaced by its substitute in a substitution box (S-Box). In cryptography, an S-box is a basic component of symmetric key algorithms used to obscure the relationship between the plaintext and the ciphertext. The next one is the ShiftRows step where the rows are cyclically shifted over different offsets. The next step is the MixColumns, where each column is multiplied with a matrix over the Gallois Field, denoted as GF(2^8). The last step of the round operation is another AddRoundKey. It is a simple XOR with the actual data and the subkey for the current round. Before producing the final ciphered data Y_i, the AES performs an extra final routine that is composed of (SubBytes, ShiftRows and AddRoundKey) steps as shown in Fig. 4.3.

The process over the plaintext data X_i is independent of the process over the secret key, and is called KeySchedule. It is made up of two components: the KeyExpansion and the RoundKeySelection. The ExpandedKey is a linear array of 4-byte words and is given by $W[N_b \times (N_k + 1)]$, where N_b is the number of columns of the data block and N_k is the number of columns of the cipher key. The first N_k words contain the cipher key and all of the other words are defined recursively. The KeyExpansion function depends on the value of N_k while the cipher key is expanded into an ExpandedKey.

4.3.2 Working modes for AES

AES cipher can support several cipher modes *e.g.*, ECB (electronic code book), CBC (cipher block chaining), OFB (output feedback), CFB (cipher feedback), CTR (Counter)[Stinson 2005] and others. The ECB mode is actually the basic AES algorithm. With the ECB mode, each plaintext block X_i is encrypted with the same secret key k producing the ciphertext block Y_i:

$$Y_i = E_k(X_i), \tag{4.1}$$

where $E_k(.)$ is the block encryption function. The CBC mode adds a feedback mechanism to a block cipher. Each ciphertext block Y_i is XORed with the incoming plaintext block X_{i+1} before being encrypted with the key k. An initialization vector (IV) is used for the first iteration. In fact, all modes (except the ECB mode) require the use of an IV.

In CFB mode, Y_0 is substituted by the IV as shown in Fig. 4.4. The keystream element Z_i is then generated and the ciphertext block Y_i is produced as:

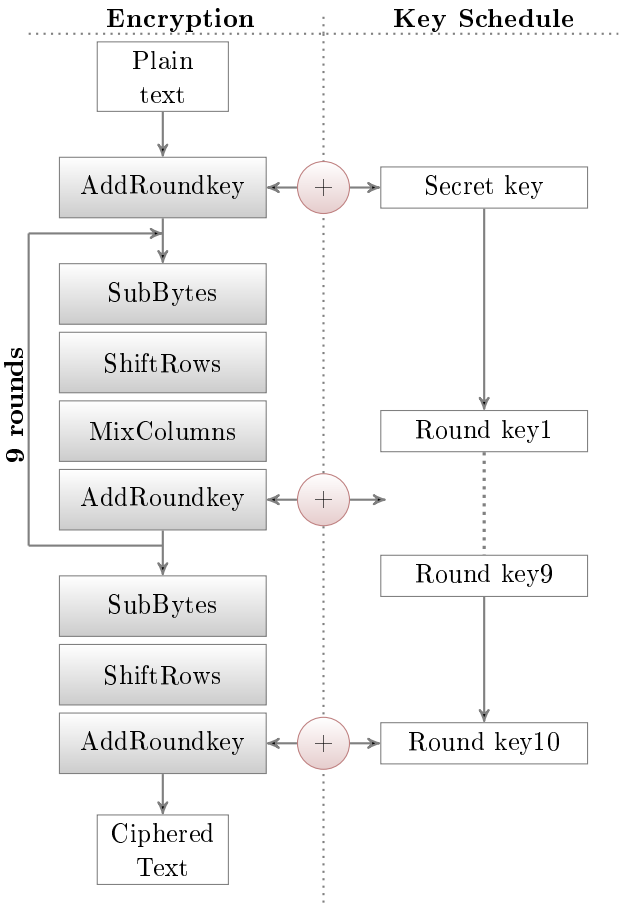


Figure 4.3: Block diagram of AES encryption algorithm containing 9 rounds of processing steps.

$$\begin{cases} Z_i &= E_k(Y_{i-1}), \text{for } i \geqslant 1 \\ Y_i &= X_i \oplus Z_i \end{cases}, \tag{4.2}$$

where $\oplus$ is the XOR operator.

In the OFB mode, Z_0 is substituted by the IV and the input data is encrypted by XORing it with the output Z_i.

The CTR mode has very similar characteristics to OFB, but in addition it allows pseudo-random access for decryption. It generates the next keystream block by encrypting successive values of a counter.

Although AES is a block cipher, in the OFB, CFB and CTR modes it operates as a stream cipher. These modes do not require any special measures to handle messages whose lengths are not multiples of the block size since they all work by XORing the plaintext with the output of the block cipher. Each mode has its advantages and disadvantages. For example in ECB and OFB modes, any modification in the plaintext block X_i causes the corresponding ciphered block Y_i to be altered,

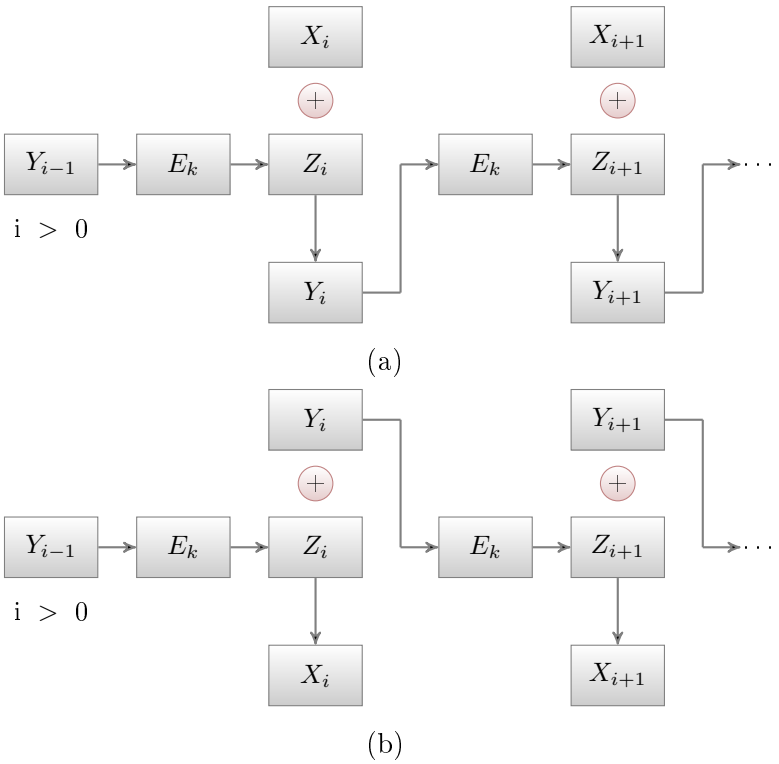


Figure 4.4: Cipher feedback mode of block cipher: (a) Encryption process, (b) Decryption process.

but other ciphered blocks are not affected. On the other hand, if a plaintext block X_i is changed in CBC and CFB modes, then Y_i and all subsequent ciphered blocks will be affected. These properties mean that CBC and CFB modes are useful for the purpose of authentication while ECB and OFB modes treat separately each block. Therefore, we can notice that OFB does not spread noise, while the CFB does exactly that.

Modern encryption algorithms including AES take lot of processing power. They are not suitable for encryption of multimedia mainly for two reasons. First, multimedia data is huge in size and full encryption demands lot of processing power. Second, full encryption will also encrypt the header information, which is critical for certain operations. This limitation gives rise to introduction of low-processing techniques which do not take lot of processing power but protects the visual data by degrading the quality *e.g.*, selective encryption (SE), partial encryption (PE) and soft encryption. In Section 4.4, we are presenting an overview of SE and PE.

4.4 Selective and partial encryption of multimedia data

Digital rights management (DRM) systems enforce the rights of the multimedia property owners while ensuring the efficient rightful usage of such property. Multi-

media data requires either full encryption or selective encryption depending on the application requirements. For example military and law enforcement applications require full encryption. Nevertheless, there is a large spectrum of applications that demand security on a lower level, as for example that ensured by selective encryption (SE) or partial encryption (PE). Such approaches reduce the computational requirements in networks with diverse client device capabilities [Cheng 2000].

Selective encryption encrypts some part of the data to get desired level of security and utilizes minimal computational resources. While in partial encryption, we encrypt only a portion of image (*e.g.*, ROI), rather than encryption of the whole image. SE and PE can be used to process and transmit the visual content in real-time. The security level of PE or SE is always lower when compared to the full encryption. On the other hand PE or SE decreases the data size to be encrypted and consequently requires lower computational time which is an important asset in wireless and portable multimedia systems. In this case we have a trade off between the amount of encrypted data and the necessary computational resources.

Confidentiality is very important for low powered embedded systems such as for example smart phones. When considering image processing applications on such devices we should use minimal resources. The modern ciphers are usually too slow to be used for image and video processing in commercial low powered systems. The selective encryption (SE) can fulfill the application requirements without the computational overhead of the full encryption. In case of SE, only the minimum necessary data are ciphered. However, the security of SE is always lower when compared to that of the full encryption. Substantial computational reduction is one of the main advantages of SE and PE. The security of partially encrypted videos to attacks which exploit the information from non-encrypted bits together with the availability of side information has been studied in [Said 2005].

Selective encryption can be format compliant and can leave important header information without encryption. Format compliance is a required feature in many applications such as rate adaptation for multimedia on heterogeneous networks [Li 2001], DC image extraction for multimedia content searching [Yeo 1995]. It is also required for bandwidth adaptation [Wu 2000], unequal error adaptation [Wang 2000] and random access [Wen 2002]. Moreover, the relation between the quality and timeliness is very important. For example, a football match requires protection during the show to ensure revenue from the viewers, but the demand of security reduces or even vanishes over the following days.

4.5 Prior work on SE and PE of image and video data

Selective encryption (SE) is a technique aiming to save computational time or to enable new system functionalities by only encrypting a portion of a compressed bitstream while still achieving adequate security [Lookabaugh 2004]. SE and partial encryption (PE) are applied only on certain parts of the bitstream. In the decoding stage, both the encrypted and the non-encrypted information should be appropri-

ately identified and displayed [Cheng 2000, Martin 2005, Rodrigues 2006].

The technical challenges posed by DRM systems are high and previous approaches have not entirely succeeded in tackling them [Lin 2005]. The protection rights of individuals and the privacy of certain moving objects in the context of security surveillance systems using viewer generated masking and the AES encryption standard has been addressed in [Yabuta 2005].

Different encryption techniques including permutation, DES and AES have been used for SE of image and video in literature. Theoretically, video codec can be interrupted at any point to perform encryption. But the candidate domains for SE can be divided into five broad categories namely spatial, video codec structure, transform, entropy coding stage and bitstream as shown in Fig. 4.5. In this section, the state of the art of SE and PE techniques for image/video is presented in the following subsections.

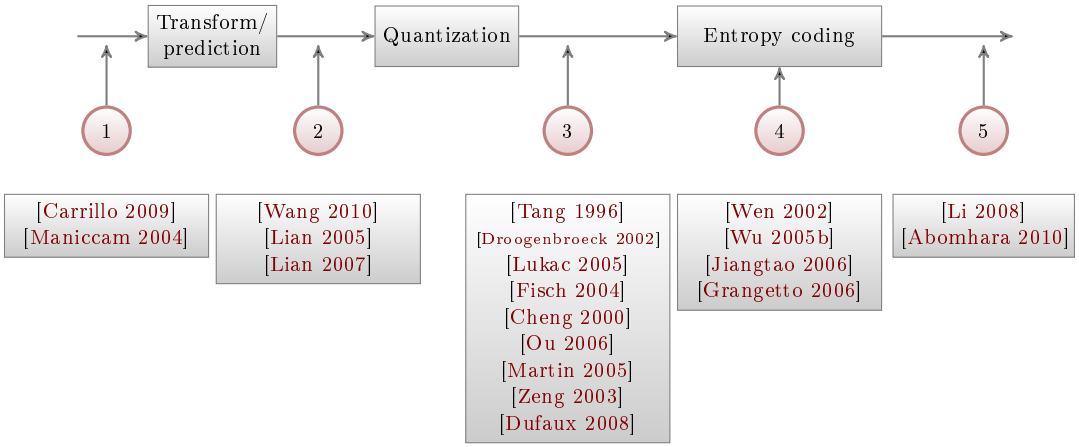


Figure 4.5: Encryption of video data before, during and after compression at different stages: (1) Spatial, (2) Video codec structure, (3) Transform domain, (4) Entropy coding stage, (5) Bitstream.

4.5.1 Spatial domain encryption

Spatial domain is a straight forward place to apply generic encryption to multimedia as shown by stage 1 in Fig. 4.5. There is increase in bitrate for such methods because encryption increases the entropy of the visual data. The main advantage of these schemes is that they are independent of any compression standards and if they are robust against compression, they can also withstand video transcoding attacks. Spatial domain SE of H.264/AVC has been discussed in [Carrillo 2009] wherein they do permutations of the pixels of MBs which are in ROI. The drawback of this scheme is that bitrate increases as the size of ROI increases. This is due to increase in entropy of ROI region.

Another technique which encrypts video pixels by using different SCAN patterns

was proposed by Manicaam and Bourbakis [Maniccam 2004]. They took difference of frames and encrypted the residual data. The drawback of this method is that the bitrate of video bitstream will rise by several times.

4.5.2 Encryption in the video codec structure

Instead of performing SE in video data (e.g. pixels, coefficients, entropy codes), some researchers have also proposed the encryption of header data. At stage 2 in Fig. 4.5, all of this header information is available for encryption.

In [Wang 2010], Wang *et al.* have presented a SE scheme for H.264/AVC. They proposed to encrypt *intra* prediction mode, motion vector difference, and quantization coefficients. A hierarchical key generation scheme is also proposed, in which the encryption keys are generated based on the cryptographic hash function. Since generated frame keys have been consistent with the corresponding frame numbers, frames are always synchronized in the decrypting process even if the frame loss occurs. Hence this scheme is secure against some frame synchronization attacks.

In [Lian 2005], Lian *et al.* have performed SE in some fields like *intra* prediction mode, residual data, *inter* prediction mode and motion vectors. A scheme for commutative encryption and watermarking of H.264/AVC is presented in [Lian 2007]. Here SE of some MB header fields is combined with watermarking of magnitude of DCT coefficients. This scheme presents a watermarking solution in encrypted domain without exposing video content.

The limitation of these techniques is that encrypted video bitstream will not be fully format compliant. If video frame headers are encrypted, the bitstream will be recognized as video bitstream but it will not be browse-able at frame level. If only MB header data are encrypted, the video bitstream cannot be parsed browse-able at MB level. In case the system level headers are encrypted. The bitstream is not format compliant at all.

4.5.3 Transform domain encryption

Video data can also be encrypted in transform domain as shown by stage 3 in Fig. 4.5. Several transform domain SE techniques have been proposed in the literature.

In [Tang 1996], Tang proposed a technique called scan permutation applicable to DCT-based image and video codecs. This method provides a certain level of confidentiality, but it increases the overall bitrate. In [Droogenbroeck 2002], Droogenbroeck and Benedett proposed a technique for encryption of JPEG images. It encrypts a selected number of AC coefficients. The DC coefficients are not ciphered since they carry important visual information and they are highly predictable. In spite of the constancy in the bitrate while preserving the bitstream compliance, the compression and the encryption process are separated and consequently the computational complexity is increased. In [zeng2003], Zeng and Lei proposed to shuffle quantized transform coefficients across DCT blocks, but within the same frequency

band.

Combining PE and image/video compression using the set partitioning in hierarchical trees was used in [Cheng 2000]. Nevertheless, this approach requires a significant computational complexity. A method that does not require significant processing time and which operates directly on the bit planes of the image was proposed in [Lukac 2005].

The AES was applied on the Haar discrete wavelet transform compressed images in [Ou 2006]. The encryption of color images in the wavelet transform has been addressed in [Martin 2005]. In this approach the encryption is performed on the resulting wavelet code bits. Fisch *et al.* [Fisch 2004] proposed a scalable encryption method for a DCT-coded visual data wherein the data are organized in a scalable bitstream form. These bitstreams are constructed with the DC and some AC coefficients of each block which are then arranged in layers according to their visual importance and PE process is applied over these layers.

In [Zeng 2003], SE of MPEG4 video standard is proposed by doing frequency domain selective scrambling, DCT block shuffling and rotation. This scheme is very easy to perform but its limitation is its bitrate overhead. SE of ROI of MPEG4 video has been presented in [Dufaux 2008]. It performs SE by pseudorandomly inverting sign of DCT coefficients in ROI.

4.5.4 Encryption in the entropy coding module

Encryption during the entropy coding module has been investigated by several authors. It is shown by stage 4 in Fig. 4.5.

The use of Huffman entropy coder as encryption cipher has been studied in the literature in [Wen 2002, Wu 2005b]. Entropy coding based SE of MPEG4 video standard has been studied in [Wen 2002] wherein Data Encryption Standard (DES) was used to encrypt fixed length and variable length codes. In this approach, the encrypted bitstream is completely compliant with MPEG4 bitstream format but it increases the bitrate. A trade off has to be made among complexity, security and the bit overhead. In [Wu 2005b], Wu and Kuo have presented an encryption technique based on statistical models. They proposed to perform encryption by using different Huffman tables for different input symbols. The tables, as well as the order in which they are used, are kept secret. This technique is vulnerable to known plaintext attacks as explained in [Jakimoski 2008].

The conversion of arithmetic entropy coding module into encryption cipher has been proposed in [Jiangtao 2006, Grangetto 2006]. In [Jiangtao 2006], key-based interval splitting of arithmetic coding (KSAC) has been proposed wherein intervals are partitioned in each iteration of arithmetic coding. Secret key is used to decide how the interval will be partitioned. Number of sub-intervals in which an interval is divided should be kept small as it increases the bitrate of bitstream. Randomized arithmetic coding [Grangetto 2006] is aimed at arithmetic coding but instead of partitioning of intervals like in KSAC, secret key is used to scramble the order of intervals. The limitation of entropy coding based techniques is that encrypted

bitstream is not format compliant. Moreover, arithmetic coding based techniques require lot of processing power.

4.5.5 Encryption in the video bitstream

Encryption of the compressed video is explored by few researchers. It is shown by stage 5 in Fig. 4.5. SE of H.264/AVC at network abstraction layer (NAL) has been proposed by [Li 2008]. Important NAL units namely instantaneous decoding refresh (IDR) picture, sequence parameter set (SPS), and picture parameter set (PPS) are encrypted with a stream cipher. This scheme is not format compliant and cannot be parsed even at frame level. SE of H.264/AVC using block-based AES has been proposed in [Abomhara 2010]. In this scheme, encryption of I frame is performed, since P and B frame have no significance without I frames. The limitation of SE at this stage is that network level decisions (*e.g.*, unequal error protection, network adaptive bit allocation etc.) cannot be taken, since video header information is also encrypted.

4.6 Summary

In this chapter, we discussed basic principles of encryption along with prior work in selective and partial encryption of visual content. Before presenting the state of the art, we presented the basic techniques of cryptography along with their classification. Being the latest and having the highest security, we discussed working of AES in detail followed by the operational modes with an emphasis on CFB mode. The concept of selective and partial encryption of multimedia content is then presented. It is followed by an elaborate discussion on state of the art of selective and partial video encryption techniques with a focus on H.264/AVC. This chapter will serve as a technical foundation for our work on SE of H.264/AVC.

Chapter 5

Image and Video Fingerprinting

Contents

5.1 Introduction

Tandis que le tatouage est utilisé pour la protection des droits d'auteur et le crypage pour l'accès limité, la prise des empreintes digitales active est utilisée pour tracer les utilisateurs malhonnêtes en cas de distribution illégale. Dans la prise des empreintes digitales, un code de prise des empreintes digitales séparé, permettant l'identification d'un utilisateur, est incorporé dans la copie personnelle de chaque utilisateur en utilisant une technique de tatouage robuste. Si un utilisateur naïf distribue une copie de son contenu avec ses empreintes digitales, donc la copie piratée peut facilement être tracée et il est posible de remonter à l'utilisateur coupable. Le traçage de l'utilisateur coupable devient plus difficile quand un ensemble d'utilisateurs, appelés des pirates, forme une coalition pour détecter les empreintes digitales et les modifier/effacer avant la distribution illégales des données. Ces attaques sont appelées des attaques de collusion et peuvent être linéaires ou non-linéaires. Les codes de prise des empreintes digitales doivent être résistants à ce type d'attaque.

Des explications présentent des attaques de collusion linéaires et non-linéaires dans ce chapitre. Il est accompagné par des modèles différents du problème de prise des empreintes digitales. Les codes d'anti-collusion sont présentés avec un accent sur les codes de Tardos.

While watermarking is used for copyright protection and encryption is used for restricted access, active fingerprinting is used for tracing the dishonest users in case of illegal distribution. In fingerprinting, a separate fingerprinting code, identifying a user, is embedded in the personal copy of each user using a robust watermarking technique. If a naive user distributes a copy of his fingerprinted content, then the pirated copy can easily be traced back to the guilty user and hence he will be exposed. Tracing the guilty user becomes more difficult when a collection of users called pirates form a coalition to detect the fingerprints and modify/erase them before illegally distributing the data. These attacks are called collusion attacks and can be linear or non-linear. Fingerprinting codes must be resistant to such type of attacks. Both linear and non-linear collusion attacks have been explained in this chapter. A fingerprinting code is designed for each user in a way that enables the distributor to identify **at least one of the pirates** as long as the coalition size does not exceed a certain threshold c, which is one of the design parameters for design of a fingerprinting code. Basic classification and construction of fingerprinting codes is presented in this chapter. It is accompanied by different models of a fingerprinting problem. Anti-collusion codes are presented with an emphasis on Tardos fingerprinting codes.

This chapter has been arranged as follows. In Section 5.2, we present the classification and underlying assumptions used to model the fingerprinting problem in literature. Practical fingerprinting codes are presented in Section 5.3 with a focus on Tardos fingerprinting codes. It is followed by prior work on fingerprinting codes in Section 5.4. Section 5.5 explains the linear and non-linear collusion attacks and is followed by concluding remarks in Section 5.6.

5.2 Fingerprint code construction

5.2.1 Classification

Consider a code $\mathcal{C}$ of length m on an alphabet Q with $|Q| = q$ and $\mathcal{C} \subseteq Q^m$. Elements of $\mathcal{C}$ are called codewords here. Each codeword has the form $x_1, \ldots, x_m)$, where $x_i \in Q$, $1 \leq i \leq m$.

For any subset of codewords $\mathcal{C}_t \subseteq \mathcal{C}$, we define the set of **descendants** of $\mathcal{C}_0$ denoted $desc(\mathcal{C}_0)$ by:

$$desc(\mathcal{C}_0) = \{x \in Q^m : x_i \in \{a_i : a \in \mathcal{C}_0\}, 1 \leq i \leq m\}.$$

The set $desc(\mathcal{C}_0)$ consists of the m-tuples that could be produced by a coalition holding the codewords in the set $\mathcal{C}_0$. Now let c be a positive integer. For a code $\mathcal{C}$,

define the *c-descendant code* of $\mathcal{C}$, denoted $desc_c(\mathcal{C})$, as follows:

$$desc_c(\mathcal{C}) = \bigcup_{\mathcal{C}_0 \subseteq \mathcal{C}, |\mathcal{C}_0| \leq c} desc(\mathcal{C}_0)$$

The set $desc_c(\mathcal{C})$ consists of the m-tuples that could be produced by some coalition of size at most c. We now give the following definitions concerning traceability properties of codes.

- *frameproof (FP):* A code is c-FP if no coalition of size at most c can frame another user not in the coalition by producing the codeword held by that user. Hence $\mathcal{C}$ is a c-FP code provided that for all $x \in desc_c(\mathcal{C})$, $x \in desc(\mathcal{C}_i) \cap \mathcal{C}$ implies $x \in \mathcal{C}_i$.

- *secure-frameproof(SFP):* A code is c-SFP if no coalition of size at most c can frame a disjoint coalition of size at most c by producing an m-tuple that could have been produced by the second coalition. Hence $\mathcal{C}$ is a c-FP code provided that for all $x \in desc_c(C)$, $x \in desc(\mathcal{C}_i) \cap desc(\mathcal{C}_j)$ implies $\mathcal{C}_i \cap \mathcal{C}_j \neq \varnothing$.

- *identifiable parent property (IPP):* A code is c-IPP if no coalition of size at most c can produce an c-tuple that cannot be traced back to at least one member of the coalition. In other words, $\mathcal{C}$ is c-IPP code provided that for all $x \in desc_w(\mathcal{C})$, it holds that:

  $\bigcap_{\{i:x \ \in \mathrm{desc}(\mathcal{C}_i)\}} \mathcal{C}_i \neq \varnothing$.

- *c-tracibility (TA):* c-TA codes are those c-IPP codes that allows a linear time algorithm to determine an identificable parent. For $x, y \in Q^m$, define $I(x, y) = \{i : x_i = y_i\}$. $\mathcal{C}$ is a c-TC (traceability) code provided that, for all i and for $x \in desc(\mathcal{C}_i)$, there is at least one codeword $y \in \mathcal{C}_i$ such that $|I(x, y)| > |I(x, z)|$ for any $z \in \mathcal{C}$.

The relationship among all the four code categories can be given as:
c-FP $\subset$ c-SFP $\subset$ c-IPP $\subset$ c-TA

IPP codes are deterministic codes with (exactly) zero error probability and was considered independently by Hollmann *et al.* [Hollmann 1998]. They were further studied in [Barg 2001, Barg 2004, Alon 2004, Blackburn 2003] among others.

5.2.2 Fingerprinting model

The problem of active fingerprinting can be explained with Fig. 5.1. As an example, if users 2 and 3 collaborate to create the pirated copy, the objective of the distributor will be to output either 2 or 3 using a decoding algorithm. The distributor commits a decoding error if he is unable to identify any user as a member of the pirate coalition (called false negative) or if a user outside the coalition is identified as a pirate (called false positive). The design parameters for a fingerprinting code include its code length m, total number of users n, maximum number of colluders c_0, false-positive (ε_1) and false-negative (ε_2) values.

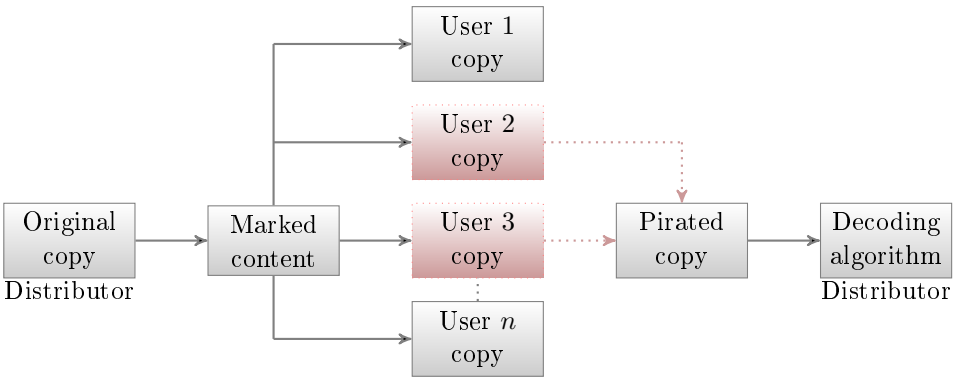


Figure 5.1: Basic collusion attack model for fingerprinting. User 2 and user 3 collude their copies to create a colluded copy.

There are two main setups considered for the fingerprinting problem in the literature namely, the distortion setting and the marking assumption. In this section, we are presenting an overview of both of these fingerprinting models.

5.2.2.1 Distortion setting

The distortion setting is commonly used in applications relating to multimedia fingerprinting [Moulin 2005, Trappe 2003, Wu 2004, He 2006]. In this model, the fingerprint is usually an *additional signal* which is superimposed on the original *host* data in such a way that the difference, or distortion, between the original and the fingerprinted copies is smaller than some threshold. The coalitions are restricted to creating a forgery which has distortion less than some threshold from at least one of the colluders' fingerprinted copies.

In the distortion setting, some results on the information theoretic limits were known (see [Moulin 2003, Somekh-Baruch 2005, Somekh-Baruch 2007, Somekh-Baruch 2008]). In particular, Somekh-Baruch and Merhav [Somekh-Baruch 2005, Somekh-Baruch 2008] obtained upper and lower bounds for the optimal rate which differ by a factor corresponding to the maximum coalition size. However, the methods used to establish these bounds are insufficient to establish tight bounds on the capacity of the digital fingerprinting problem or to construct code families and decoding algorithms for it.

5.2.2.2 Marking assumption

The line of research into the construction of fingerprinting schemes followed in this manuscript applies to systems designed to protect digital content and is based on the following mathematical model:

1. A *content* is a sequence of symbols. Here, we consider a binary alphabet.

2. In the original *content*, there are m locations which we can change the symbol without significantly degrading the content. The codeword, a binary sequence of length m identifying a user will be hidden in the content to such sites.

3. The pirates do not know *a priori* fingerprint positions. The pirate coalition attempts to uncover some of the fingerprint positions by comparing their copies of the data for differences. once such a difference is located in some position, it is guaranteed to be a fingerprint position. The comparison procedure does not reveal any information regarding the bits that are identical in all the data copies of the coalition members, which can be either information or fingerprint digits.

4. The pirated copy is created by modifying only those positions which are different in the fingerprinted copies of the pirates.

5. The process of accusation knows these places and samples the pirated copy of a sequence, called pirate sequence of m symbols.

It is important to mention that the original unmarked object is never distributed or released. This is especially true with digital content, since simple binary comparison on a marked object versus the original will reveal the locations of all marks even when the marks are unperceivable by human visual system.

5.2.3 Why probabilistic codes?

The design objective of a fingerprinting system is to provide a code construction that can trace at least one of the colluders with zero error. Unfortunately, for user $n \geq 3$, and colluders $c \geq 2$, no deterministic solution is possible and one has to opt for probabilistic solutions, allowing some low probability of error.

It is known [Boneh 1998] that any fixed assignment of fingerprints (a deterministic code) cannot satisfy this requirement: namely, there exist several attacking strategies of the coalition that will result in the error probability bounded away from zero irrespective of the decoding employed. For this reason it becomes necessary for the distributor to use probabilistic codes, where the random key is known only to the distributor.

In a probabilistic fingerprinting code, the following parameters are important:

- n: The number of users.
- c: the number of colluders.
- ε_1: False positive probability.
- ε_2: False negative probability.
- m: all the above four parameters result in a fingerprint code of certain length denoted by m.

The code length m influences to great extent the practical usability of a fingerprinting scheme, as the number of segments m that can be used to embed a fingerprint symbol is severely constrained; typical video watermarking algorithms for instance can only embed seven bits of information in a robust manner in one minute of a video clip [Standard 2007]. Furthermore, distributors are interested in the shortest possible codes that are secure against a large number of colluders, while accommodating a huge number n of users (of the order of $n = 10^6$ or even $n = 10^9$).

Low error probabilities are another central requirement. The most important type of error is accusation of an innocent user called false-positive. It is denoted by ε_1. The probability of such an event must be extremely small; otherwise the distributor's accusations would be questionable, making the whole fingerprinting scheme unworkable. The second type of error is the false-negative, where the scheme fails to accuse any of the colluders. It is denoted as ε_2. In practical situations, fairly large values of ε_1 can be tolerated. Often the objective of fingerprinting is to deter unauthorized distribution rather than to prosecute all those responsible for it. Even a mere 50 % probability of getting caught can be a significant deterrent for colluders.

5.2.4 Fingerprinting codes and lower bounds

Mathematically speaking, a fingerprint is a finite string over some q-ary alphabet $\sum$; the set of all fingerprints is called a fingerprinting code. The rate of a fingerprinting code R quantifies the trade off between the number of users that can be supported by the system and the fingerprint length required to make it workable. The rate is given by the ratio of the logarithm of the number of licensed users n to the length of the fingerprint code m:

$$R = \frac{log_{\Sigma} n}{m} \tag{5.1}$$

A code is called collusion-resistant against a coalition of size c_0, if any set of $c \leq c0$ colluders is unable to produce an untraceable copy. A fundamental question in understanding the fingerprinting problem is as follows: Given the maximum number of colluders, what is the largest rate attainable by fingerprinting codes such that there is a tracing algorithm that makes an error with an arbitrarily small probability?

Though the initial works on fingerprinting date back to the 1980s [Wagner 1983, Blakley 1985], it was the work of Chor *et al.* [Chor 1994, Chor 2000] on traitor tracing for broadcast encryption and of Boneh and Shaw [Boneh 1998] on collusion-secure fingerprinting codes that brought it to the attention of the research community. [Boneh 1998] also was the first to give a construction of code families of increasing length with vanishing error probability. They proposed such codes of length $O(c^4 \log(m/\varepsilon)\log(1/\varepsilon)$ for m users that are ε-secure against c pirates. This translates to a t-secure fingerprinting scheme with rate of $\Omega(1/c^4)$. The same paper gave a lower bound of $\Omega(t\log(1/(c\varepsilon)))$ for the length of fingerprint code with the same parameters. As this bound does not depend on the number of users it does not give an upper bound on the rates of c-secure fingerprint schemes.

5.3 Practical fingerprinting code

In literature, several fingerprinting algorithms have been proposed for images. In [Trappe 2003], Anti Collusion Codes (AAC) were presented for multimedia content, which were based on combinatorial theories with a joint coding and embedding framework. The code is derived from balanced incomplete block design (BIBD), which is then modulated by Gaussian orthogonal spreading sequences.

In [He 2006], ECC-based forensic code was proposed which uses Gaussian sequences to modulate symbols in the codeword along with additive spread spectrum embedding. ECC-based codes are the concatenation of an inner code and an outer code. By saving the inner codes and outer codes, we can easily construct the forensic codes for all the users. In [Lin 2009], Lin *et al.* have proposed improved ECC-based anti-collusion codes for images which consume less resources than Tardos fingerprinting codes.

Tardos [Tardos 2003] proposed an optimal probabilistic binary asymmetric code that reaches the lowest known bound. Before discussing in detail the state of the art, it is pertinent to explain Tardos code construction in Section 5.3.1, followed by its limitations in Section 5.3.2.

5.3.1 Tardos fingerprinting code generation

Tardos fingerprinting code is a probabilistic fingerprinting code. It can be generated in the following three steps:

- For a code of length m, we generate random and independent probabilities $\{p(i)\}_{1\leq i\leq m}$ with the distribution $f(p) = \frac{1}{\pi\sqrt{p(1-p)}}$ for $p \in [0,1]$. Practically p is between t and $1-t$ with $t = 10^{-3}$. Hence it has high frequency on the edges as shown in Fig. 5.2.

- The next step is to generate Tardos code. For n users with m code length, it is a matrix of size $m \times n$ as given in Fig. 5.3. For the case of binary Tardos code, each line of S is filled with 0 or 1 with $Prob[S(i,j) = 1] = p(i)$. Each column is a fingerprinting code for separate user.

- For accusation process, a binary sequence z is extracted from the pirated copy and an accusation score A_j is associated with user j given as:

$$A_j = \sum_{i=1}^{m} U\left(z(i), S(i,j), p(i)\right), \tag{5.2}$$

  where function U is defined as:

$$\begin{aligned} U(1,1,p) &= \sqrt{(1-p)/p} \quad & U(1,0,p) &= -\sqrt{p/(1-p)}, \\ U(0,0,p) &= \sqrt{p/(1-p)} \quad & U(0,1,p) &= -\sqrt{(1-p)/p}. \end{aligned}$$

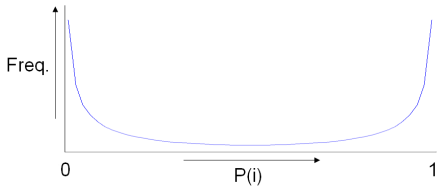


Figure 5.2: Probability distribution for the probabilities of Tardos code.

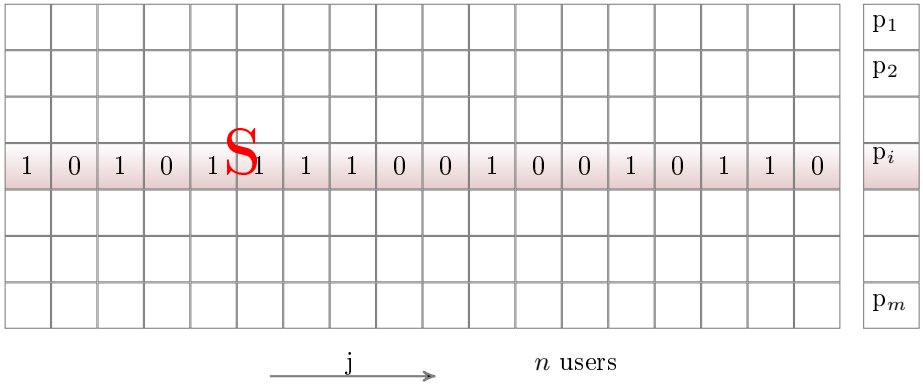


Figure 5.3: $m \times n$ Tardos code with code length of m for n users.

A fingerprinting code is analyzed based on its code length, maximum number of colluders c_0, false-positive (ε_1) and false-negative (ε_2) values. For binary asymmetric Tardos code, the length of code is given as $m = 100c_0^2 \ln(\frac{1}{\varepsilon_1})$ and ε_1 and ε_2 is given as $\varepsilon_2 = \varepsilon_1^{\frac{c_0}{4}}$ [Tardos 2003].

For accusation process, an accusation sum A_j is calculated for each user j. A_j for accused users may be modeled with a Gaussian centered at $\mu = \frac{2m}{c\pi}$, while A_j for innocent users may be modeled with a Gaussian centered at 0. Accused users (the traitors) have a score above $\mu - \sqrt{m}$ (*i.e.* $\frac{2m}{c\pi} - \sqrt{m}$), where $\sqrt{m}$ is the standard deviation of the Gaussian. A more precise threshold may be selected such that proposed in [Cérou 2008], which links ε_1 and threshold.

5.3.2 Resource requirements of Tardos fingerprinting codes

Different fingerprinting codes have different requirements of resources:

- For Tardos code, the accusation process requires $O(nm)$ computations, since we have to run the accusation process for all users. For ECC-based code, the accusation process requires only $O(\sqrt[k]{n}m)$ [He 2006] as shown in Table 5.1.

- For Tardos code, code length increases squarely with increase in the number of colluders, and logarithmically with decrease in value of ε_1 as shown in Table 5.2. For example, for $c = 10$ with $\varepsilon_1 = 10^{-4}$, the code length is 23688 bits, while it is 110540 bits for $c = 20$ with $\varepsilon_1 = 10^{-6}$.

- Since Tardos code is probabilistic in nature, we have to generate the codes for all the users separately and have to store them. Tardos code needs 64GB of memory for $n = 10^6$, $c = 5$ and $\varepsilon_1 = 10^{-3}$ with $BER = 0.32$. In contrast to it, ECC codes require 20MB of disk space as shown in Table 5.1. Memory requirements for ECC-based codes are 1/8000 to that of Tardos code.

Table 5.1: Resource comparison of Tardos code and improved ECC code [Lin 2009].

	Tardos	ECC
Code generation	5×10^11	1.3×10^8
Storage	$64GB$	$20MB$
Detection	$O(nm)$	$O(\sqrt[k]{n}m)$

Table 5.2: Tardos code length and detection threshold for different parameter settings.

n	$1 << c$	$\varepsilon_1 (<< \varepsilon_2)$	$m = 2\pi^2 c^2 [ln(1/\varepsilon_1)]$	$Z = 20c[ln(1/\varepsilon_1)]$
10^2	5	10^{-3}	3455	220
10^3	5	10^{-4}	4935	315
	5	10^{-7}	8390	535
	7	10^{-4}	9673	440
	7	10^{-7}	16443	748
	10	10^{-4}	19740	629
	10	10^{-7}	33557	1069
	20	10^{-4}	78957	1257
10^4	5	10^{-5}	5922	377
	7	10^{-5}	11687	528
	10	10^{-5}	23688	754
	20	10^{-5}	94749	1508
10^4	5	10^{-5}	5922	377
	7	10^{-5}	11687	528
	10	10^{-5}	23688	754
	20	10^{-5}	94749	1508
10^5	5	10^{-6}	6909	440
	7	10^{-6}	13542	616
	10	10^{-6}	27635	880
	20	10^{-6}	110540	1760
10^6	5	10^{-7}	8390	535
	7	10^{-7}	16443	748
	10	10^{-7}	33557	1069
	20	10^{-7}	134227	2137

5.4 Recent work for fingerprinting code

In literature, several fingerprinting algorithms have been proposed for images. Anti Collusion Code (AAC) was the first anti-collusion forensic code for multimedia content, proposed in [Trappe 2003]. It was based on combinatorial theories with a joint coding and embedding framework. The code is derived from balanced incomplete block design, which is then modulated by Gaussian orthogonal spreading sequences. In [He 2006], ECC-based forensic code was proposed which uses Gaussian sequences to modulate symbols in the codeword along with additive spread spectrum embedding. ECC-based codes are the concatenation of an inner code and an outer code. By saving the inner codes and outer codes, we can easily construct the forensic codes for all the users. In [Lin 2009], Lin *et al.* have proposed improved ECC-based anti-collusion codes for images which consume less resources than Tardos fingerprinting codes.

Tardos [Tardos 2003] constructed fingerprint codes of length at most $m = 100c^2 ln(\frac{1}{\varepsilon_1})$ for c colluders. This construction yields c-secure fingerprinting schemes of rate $1/(100c^2)$. The same paper gave lower bound of $O(c^2 log(1/\varepsilon))$ on the length of any fingerprint code with the above parameters. The constant 100 in the length $m = 100c^2 \ln (\frac{1}{\varepsilon_1})$ was subsequently improved by several papers [Skoric 2006, Skoric 2008, Blayer 2008]. All these papers go through the proof in [Tardos 2003] and optimize various parameters in the proof to improve the code length without fundamentally changing the code construction or the accusation algorithm.

Skoric *et al.* [Skoric 2006] improved the code length if the case the error parameters are in certain intervals *i.e.* the probability of accusing an innocent user is assumed to be $10^{-15} \leq \varepsilon_1 \leq 10^{-9}$ and the probability of failing to detect pirates is $0.1 \leq \varepsilon_2 \leq 0.5$. For values within these intervals, the code length is calculated experimentally for $t \leq 100$ and it is observed that when t increases the code length approximates $4\pi^2 t^2 \log m$. This represents an improvement by a factor around $2:5$.

Skoric *et al.* [Skoric 2008] make a simplifying assumption about the pirate strategy. Even though the assumption is reasonable there is no mathematical justification for it. With this simplifying assumption they achieve an improvement by a factor of almost 10.

Blayer and Tassa [Blayer 2008] rigorously extract some formulas for the parameters in the proof of [Tardos 2003]. Their experiments confirmed an improvement by a factor of more than 4 and less than 5.

Anthapadmanabhan *et al.* [Anthapadmanabhan 2007] choose a different information theoretic approach. They construct t-secure fingerprinting schemes whose rates for t = 2 and 3 are much higher than previously obtained rates but the rate of their schemes deteriorates exponentially with t. They are the first to prove upper bounds on the rates of t-secure fingerprinting schemes. The upper bounds in their paper is given in terms of a hard to evaluate information theoretic minimax formula. They estimate this formula and prove strong upper bounds on the t-fingerprinting capacity for small values of c (namely 2 and 3) and an $O(1/c)$ asymptotic bound.

Amiri and Tardos [Amiri 2009] combine two approaches represented by [Tardos 2003] and [Anthapadmanabhan 2007] to obtain fingerprinting codes of rates higher than those achieved with either method separately. Codeword construction method is very similar to that of [Tardos 2003]. The optimal distribution is found through a game-theoretic equilibrium and prove the upper bound to be $1.78/t^2 + O(1/t^2)$. On the contrary, accusation algorithm is fundamentally inspired by the work of Anthapadmanabhan *et al.* [Anthapadmanabhan 2007]. For $t = 2$, their codes coincide with the codes given in [Anthapadmanabhan 2007] but for $t \geq 3$, their rates are better than the rates of previously obtained codes. For $t \leq 8$, they prove the $R_t \geq t^{-2}/(2ln2) + O(t^{-2})$ asymptotic bound. This improves the rates of the codes in [Tardos 2003] by a factor over 72 for large values of t.

In case of multimedia, the colluders usually apply post-processing after collusion. For instance, the colluders can compress the multimedia to reduce the data size to efficiently redistribute the colluded copy. Therefore, it is important to design a collusion resistant forensic code that is robust to post-processing. In [Xie 2008], Tardos fingerprinting code has been used with zero-bit broken arrows watermarking scheme for images. They have shown that this combination has ruled out the fusion class of attacks.

5.5 Attacks against fingerprinting codes

Several types of linear and non-linear collusion attacks may be used against multimedia fingerprints. The simplest attack may be to synchronize the media signals and average them, which is an example of the linear collusion attack. Other collusion attacks may employ nonlinear operations, such as taking the maximum or median of the values of corresponding components of individual copies. Collusion attacks may also be accompanied by post-processing attacks *e.g.*, compression, noise removal etc. Fingerprinting codes provide the protection against collusion attacks, while protection against post-processing attacks is provided by robust watermarking mainly.

In the following, post processing attacks are being described in Section 5.5.1. Linear collusion attacks are being discussed in Section 5.5.2 while Section 5.5.3 explains non-linear collusion attacks.

5.5.1 Post processing attacks

Most of the proposed fingerprinting techniques assume a perfect channel between the colluded copy and the detector. However, when the protected data is multimedia, the colluders usually apply post-processing after collusion that forms an erroneous channel. For instance, the colluders can compress the multimedia to reduce the data size to efficiently redistribute the colluded copy. Such type of attacks can be performed even by a single colluder.

5.5.2 Linear collusion attacks

Linear collusion is one of the most feasible collusion attack that may be employed against multimedia fingerprints. Given several differently marked copies of the same content, the colluders linearly combine all the copies to generate a colluded copy. Linear collusion attacks can be either averaging attack or copy-and-paste attack.

5.5.2.1 Averaging collusion attack

In a K-colluder linear collusion attack, let S is a set of all K colluders and z is the colluded video. The j^{th} pixel of colluded video z_{ave} is given as:

$$z_{ave}(j) = \sum_{k \in S} \lambda_k y_k(j), \tag{5.3}$$

where the weights λ_k satisfy $\sum_{k \in S} \lambda_k = 1$ to maintain the average intensity of the original multimedia signal. With orthogonal fingerprinting, for colluder k who is assigned the weight λ_k, the energy of its fingerprint is reduced by a factor of λ_k^2. When λ_k is smaller, the trace of colluder $k's$ fingerprint is weaker and colluder k is less likely to be caught by the detector.

In multiuser collusion, since no colluder would like to take more risk than any other colluder, they usually agree to distribute the risk of being detected evenly among them and apply fair collusion. A simple way to achieve the fairness of collusion is to average all the fingerprinted signals with an equal weight for each user and let $\lambda_k = 1/K$ for all $k \in S$. Fig. 5.4 shows an example of collusion by averaging with three colluders and all three fingerprints are averaged with the same weight 1/3. In [Ergun 1999], the collusion attack was modeled by averaging the fingerprinted signals followed by an additive noise to further hinder the detection. Their work showed that $O(m/logm)$ adversaries are sufficient to defeat the underlying watermarks, where m is the total length of the fingerprint. Similar results were also presented in [Kilian 1998]. In [Su 2000], a more general linear attack was considered, where the colluders employ multiple-input single-output linear shift-invariant (LSI) filtering plus additive Gaussian noise to thwart the fingerprints. Under the assumption that all fingerprints are independent and have identical statistical characteristics, it was shown that the optimal LSI attack involves each user weighting their marked document equally prior to the addition of additive noise.

5.5.2.2 Copy-and-paste attack

Another type of fair collusion, referred to as the copy-and-paste attack, involves users cutting out portions of each of their media signals and pasting them together to form a new signal. Fig. 5.5 shows an example of the copy-and-paste attack with two colluders: Alice and Chris. The colluded copy is generated by copying the 1/3 of fingerprinted signals of Alice, Bob and Chris.

When the fingerprint mark is embedded throughout the entire host signal by such techniques as spread-spectrum and detected through correlation based detector, the

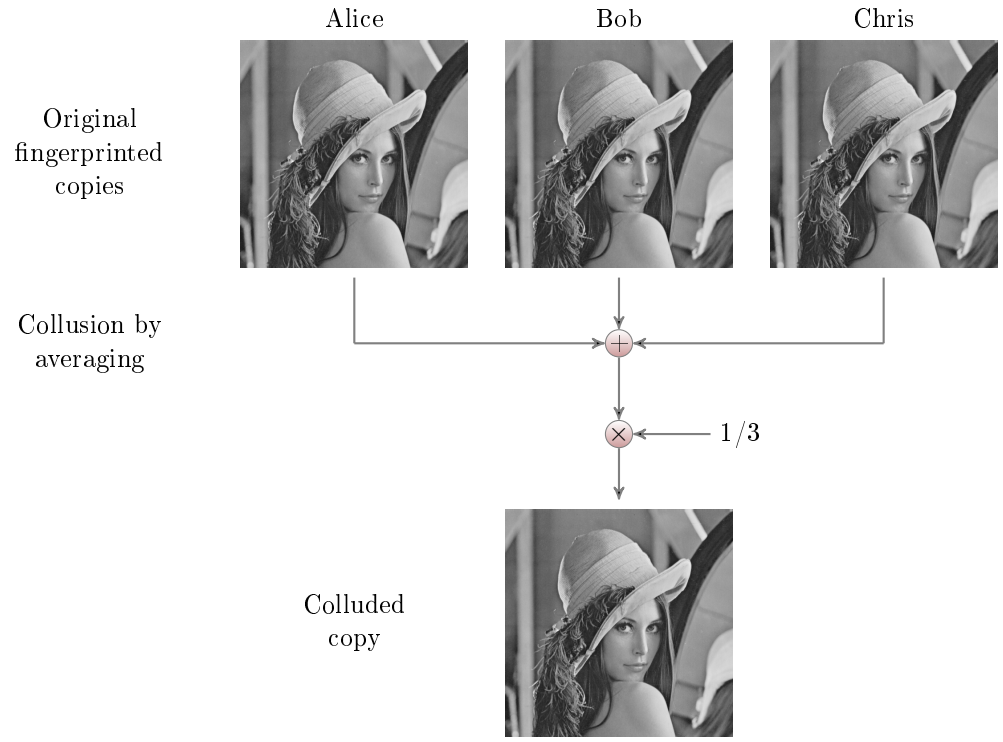


Figure 5.4: Averaging collusion: an example of linear collusion attack.

copy-and-paste collusion attack is similar to averaging collusion. In particular, in both cases, the energy of each contributing fingerprint is reduced by a factor corresponding to the amount of copies involved in the collusion. As an example, if Alice contributes 1/3 of her samples to a copy-and-paste collusion, the energy of Alice's fingerprint in the colluded copy is only 1/3 of her overall fingerprint energy. Therefore, in terms of the effect on the fingerprint energy reduction and the impact on the probability of being detected, we may consider copy-and-paste collusion analogous to average-based collusion when considering spread-spectrum embedding.

5.5.3 Non-linear collusion attacks

Linear collusion is a simple and effective way for a coalition of users to attenuate embedded fingerprints. However it is not the only form of collusion attacks available to a coalition of adversaries. In fact, for each component of the multimedia signal, the colluders can output any value between the minimum and maximum corresponding values, and have high confidence that the spurious value they get will be within the range of the just-noticeable-different since each fingerprinted copy is expected to have high perceptual quality. An important class of nonlinear collusion attacks is based upon such operations as taking the maximum, minimum, and median of corresponding components of the colluders' fingerprinted copies. For minimum/maximum/median attacks, each pixel in pirated video is the minimum,

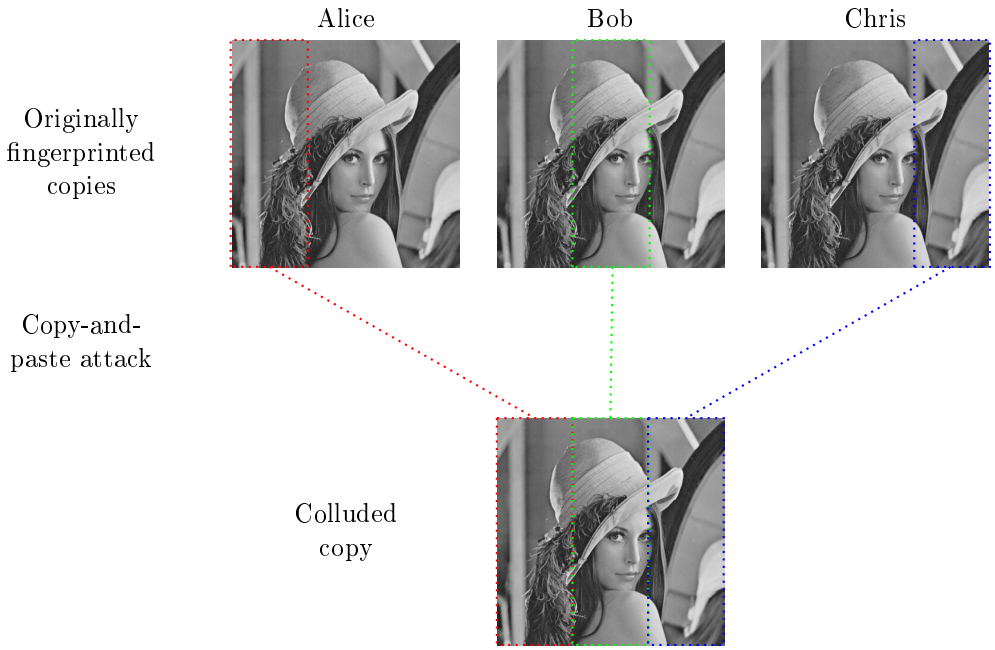


Figure 5.5: Copy-and-paste collusion: an example of linear collusion attack.

maximum, or median, of the corresponding pixels of the fingerprinted videos. For minmax attack, each colluded pixel is the average of the maximum and minimum of the corresponding pixels of the fingerprinted videos. For modified negative attack, each pixel of the attacked video is the difference between the median and the sum of the maximum and minimum of the corresponding pixels of the fingerprinted video signals. Let S is a set of all K colluders. The j^{th} pixel of colluded video $z(j)$ is given as:

$$z_{min}(j) = min\{y_k(j)\}_{k \in S} \tag{5.4}$$

$$z_{max}(j) = max\{y_k(j)\}_{k \in S} \tag{5.5}$$

$$z_{med}(j) = med\{y_k(j)\}_{k \in S} \tag{5.6}$$

$$z_{minmax}(j) = (y_{min}(j) + y_{max}(j))/2 \tag{5.7}$$

$$z_{modneg}(j) = y_{min}(j) + y_{max}(j) - y_{med}(j) \tag{5.8}$$

Fig. 5.6 gives an example of different types of nonlinear collusion and their effects with three colluders, Alice, Bob, and Chris. For one pixel at the n^{th} row and the m^{th} column in the image, assume that it takes the values 172, 173, and 176 in the three copies corresponding to the three colluders. When generating the colluded copy, for the pixel at row n and column m, the colluders can take the minimum, maximum or median of the three values, which gives 172, 176 and 173 respectively. They can also combine these basic operations to generate a colluded copy. For

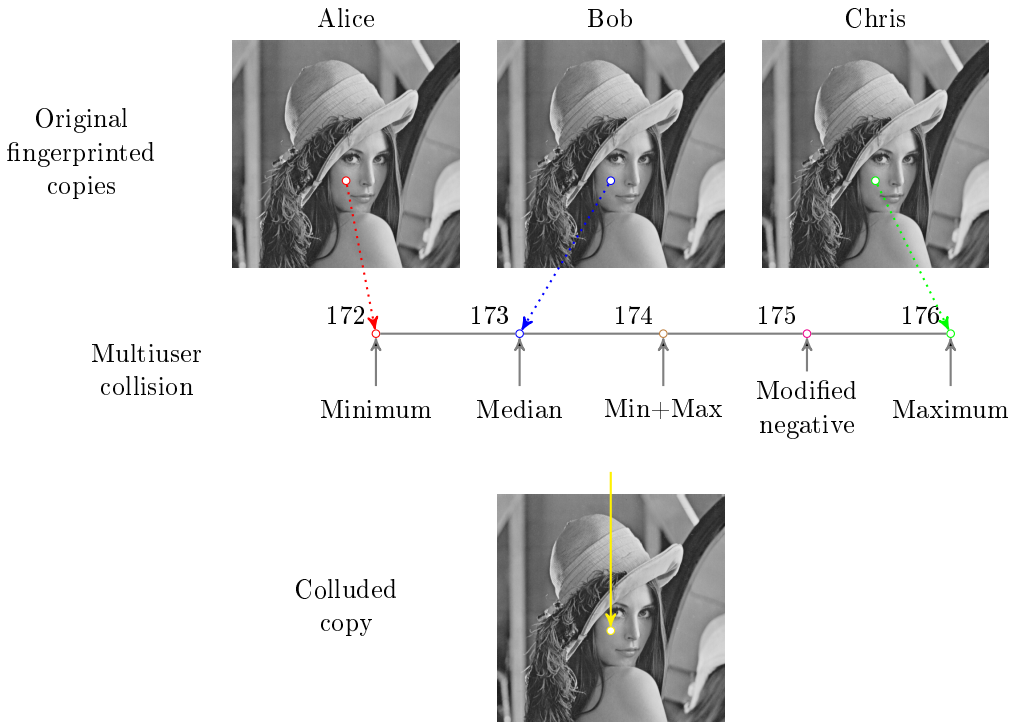


Figure 5.6: Example of non-linear collusion attacks.

example, in minmax attack, the colluders can take the average of the maximum and the minimum, which is 174 and in case of modified negative attack, the colluded pixel value will be 175. The colluders repeat this process for every pixel in the image and generate the colluded copy.

A few nonlinear attacks were studied in [Stone 1996]. For uniformly distributed fingerprints, nonlinear collusion attacks were shown to defeat the fingerprinting system more effectively than the averaging collusion [Stone 1996]. Simulation results in [Stone 1996] also showed that normally distributed fingerprints are more robust against nonlinear collusion attacks than uniform fingerprints. In addition to the robustness against collusion attacks, Gaussian fingerprints do not provide positions and the amplitudes of the embedded watermarks under statistical and histogram attacks [Craver 2004]. Therefore, Gaussian fingerprints are better than uniformly distributed fingerprints.

5.6 Summary

In this chapter, we have presented an introduction of fingerprinting codes and their use for traitor tracing. After an introduction of the domain of active fingerprinting,

we presented the distortion settings and marking assumption. It was followed by code construction and literature review of fingerprinting for image and video data. Being the latest and having the highest capacity, we discussed the working of Tardos fingerprinting code in detail, followed by its analysis and comparison with other state of the art fingerprinting codes.

With the discussion on fingerprinting of image and video data for traitor tracing, this chapter concludes the first part of this manuscript which contains the underlying concepts and state of the art in image and video protection.

Part II

CONTRIBUTIONS

Comme décrit dans le chapitre 2, des codeurs vidéo de pointe incluant H.264/AVC ont beaucoup de modes tant pour les images intra que pour les images inter. Le module de contrôleur de taux des codeurs vidéo choisit le mode qui offre le meilleur compromis entre la qualité et le débit. En insérant des données dans des vidéos, les changements de scénario et le meilleur mode de prédiction peuvent être changés. Ainsi, dans ce scénario, le contrôleur de taux doit trouver le meilleur compromis entre la qualité et le débit pour le flux de données tatoués. Le chapitre 6 se concentre sur ces questions et présente un schéma de tatouage qui prend en compte toutes ces questions, tant pour le tatouage fragile que robuste. Cela est suivi par la prise des empreintes digitales active de vidéo H.264/AVC par insertion de codes de Tardos par étalement de spectre. Dans le chapitre 7, nous nous concentrons sur le cryptage en temps réel de vidéos H.264/AVC et AVS. Puisque les données vidéos sont de tailles très importantes, nous proposons des algorithmes de cryptage sélectifs. Le flux de données crypté respecte le format original sans changement de débit pour CAVLC, CABAC et C2DVLC. Une fois que le travail de tatouage, de cryptage et d'empreintes digitales a été présenté, nous prolongeons notre travail dans le chapitre 8 en nous concentrant sur les problèmes liés aux vidéos hiérarchiques. Des algorithmes à base de parcours pour la meilleure compression et la protection des couches d'améliorations sont présentées. Dans ce travail, nous avons utilisé l'implémentation de référence de H.264/AVC JSVM 10.2 avec le mode AVC pour les expérimentations. Pour AVS, nous avons utilisé l'implémentation de référence de AVS RM 6.2c[1]*. Le contenu vidéo peut-être divers et varié. Il peut contenir différents types d'objets, de texture, de fond, de luminosité, de mouvements, etc. Neuf séquences de tests ont été utilisées pour ce travail afin de confirmer l'efficacité de nos algorithmes comme décrit en annexe A.*

As described in Chapter 2, state of the art video codecs including H.264/AVC have many modes for both *intra* and *inter* frames. The rate controller module of video codec chooses the mode which offers best quality/bitrate trade off. While inserting watermark in video data, the scenario changes and the best prediction mode may be changed. So in this scenario, the rate controller has to find the best trade off between quality, bitrate for the watermarked bitstream. Chapter 6 focuses these issues and presents a watermarking scheme which takes into account all these issues, both for fragile and robust watermarking. It is followed by active fingerprinting of H.264/AVC video by embedding Tardos fingerprinting code using spread spectrum robust watermarking technique. In Chapter 7, we focus on real-time encryption of video content for H.264/AVC and AVS. Since video data is huge in size, we propose selective encryption algorithms. The encrypted bitstream is format compliant with no change in bitrate for CAVLC, CABAC and C2DVLC. Once the basic work on watermarking, fingerprinting and selective encryption is presented, we extend our focus in Chapter 8 by concentrating on the issues related to scalable video. Scan-based algorithms for better compression and protection of

[1] http://159.22.42.57/incoming/dropbox/videosoftware/Rm62c.zip

enhancement layers of scalable video are presented. In this work, we have used the reference implementation of H.264/AVC JSVM 10.2 in AVC mode for experimental simulation. For AVS, we have used the reference implementation of AVS version RM 6.2c. Video content is very diverse in nature. It may contain different type of objects, texture, background, brightness, motion etc. Nine benchmark sequences have been used in this work to confirm the effectiveness of our algorithms as explained in Appendix A.

Chapter 6

Fragile and Robust Watermarking of H.264/AVC

Contents

6.1 Introduction

Après une présentation de l'état de l'art pour le tatouage d'image et de vidéo dans le Chapitre 3, ce chapitre présente la conception et l'analyse pour le tatouage d'images intra et inter dans le codeur vidéo H.264/AVC. Il contient aussi le tatouage d'empreintes digitales active de vidéo H.264/AVC en utilisant les codes de Tardos. La plupart des algorithmes de tatouage de vidéos prend en compte seulement des images intra pour l'insertion du message. Ceci est dû au fait que les images inter sont fortement compressées du fait de la compensation de mouvement et l'insertion du message dans ces images peut affecter significativement l'efficacité de compression. La capacité d'insertion dans les images inter est plus faibles en comparaison des images intra. Dans ce chapitre, nous analysons une méthode de

tatouage dans les images intra et inter sur la gamme entière des valeurs de QP et nous notons que la capacité d'insertion dans les images inter est comparable avec celle dans les images intra. De plus avec notre méthode, il n'y a aucun effet significatif sur le débit et le PSNR de la vidéo parce qu'au lieu d'insérer le message dans le flux compressé, nous avons incorporé celui-ci pendant le processus de codage en prenant en compte la boucle de reconstruction. Pour les coefficients non nuls, dans les images intra nous profitons du masquage spatial et dans les images inter nous obtenons naturellement l'avantage du masquage de texture et de mouvement. Il est important de mentionner que l'insertion du message n'est pas effectuée sur le flux compressé mais pendant le processus de compression et ainsi la méthode proposée prend en compte la boucle de reconstruction.

After presenting the state of the art for image and video watermarking in Chapter 3, this chapter presents design and analysis for watermarking of *intra* and *inter* frames in H.264/AVC video codec. It also contains active fingerprinting of H.264/AVC video using Tardos fingerprinting code. Most of video watermarking algorithms take into account only *intra* frames for message embedding. This is due to the perception that *inter* frames are highly compressed by motion compensation and embedding message in them may affect the compression efficiency significantly. Payload of *inter* frames is also perceived to be less as compared to *intra* frames because of lesser residual data. In this chapter, we analyze watermarking in *intra* and *inter* frames over the whole range of QP values and we note that payload of *inter* is comparable to that of *intra* frames. There is no significant effect over the overall bitrate and PSNR of the video bitstream because instead of embedding message in the compressed bitstream, we have embedded it during the encoding process by taking into account the reconstruction loop. For the non-zero QTCs, in *intra* frames we are benefited from the spatial masking and in *inter* frames we naturally get advantage of motion and texture masking. It is important to mention that watermark message embedding is not performed on the compressed bitstream rather it is carried out during the compression process and thus the proposed scheme takes into account the reconstruction loop.

This chapter is organized as follows. Section 6.2 presents the comparison of embedding after and within the reconstruction loop, while Section 6.3 contains the proposed technique for fragile watermarking of H.264/AVC. Fingerprinting of H.264/AVC video is presented in Section 6.4, wherein Tardos fingerprinting code is embedded using Spread Spectrum embedding in H.264/AVC video. Concluding remarks are presented in Section 6.5.

6.2 Reconstruction loop and message embedding

In video codec, there is a need to reconstruct the compressed video content for future prediction process. Watermark message embedding can be performed either outside this reconstruction loop (i.e. just before the entropy coding) or inside the

reconstruction loop. Comparison of embedding after and within the reconstruction loop is performed in Section 6.2.1. Hidden message aware rate-distortion because of taking into account the reconstruction loop is explained in Section 6.2.2.

6.2.1 Comparison of embedding after and within the reconstruction loop

Message embedding process in QTCs before entropy coding is analogous to embedding a message in a compressed bitstream as shown in Fig. 6.1. This includes two watermarking approaches. The first approach embeds message in VLC domain and bitstream is only to be entropy decoded to use this approach e.g. as proposed by Lu *et al.* [Lu 2005]. Another approach embeds message in DCT domain and for this approach, bitstream has to be entropy decoded and inverse quantized. Example of this approach is differential energy watermarking scheme proposed by Langelaar *et al.* [Langelaar 2001].

Embedding message after reconstruction loop creates two problems. First, we start reconstruction on the encoder side with QTC $\hat{y}(u,v)$ while on the decoder side we start decoding with watermarked QTC $\hat{y}_w(u,v)$. This results in a mismatch on the decoder side, which keeps on increasing because of the prediction process. Because of this mismatch, the difference in PSNR is very significant even for *intra* frames, let alone the *inter* frames. Second, Rate Distortion (RD) bit allocation algorithm works in quantization module and any change in quality/bitrate trade off because of the watermarking of QTCs is not taken into account.

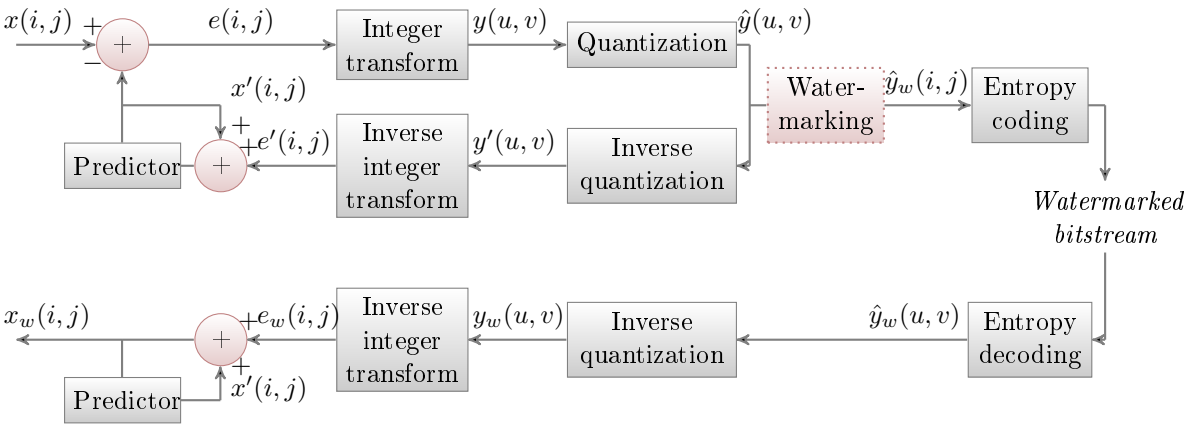


Figure 6.1: Block diagram of H.264/AVC along with the watermarking module outside the reconstruction loop.

To solve both the problems, message embedding should be performed inside the reconstruction loop as shown in Fig. 6.2. In this case, we have the same watermarked QTC $\hat{y}_w(u,v)$ on both encoder and decoder side for prediction and RD bit allocation algorithm works on $\hat{y}_w(u,v)$ for both *intra* and *inter* frames.

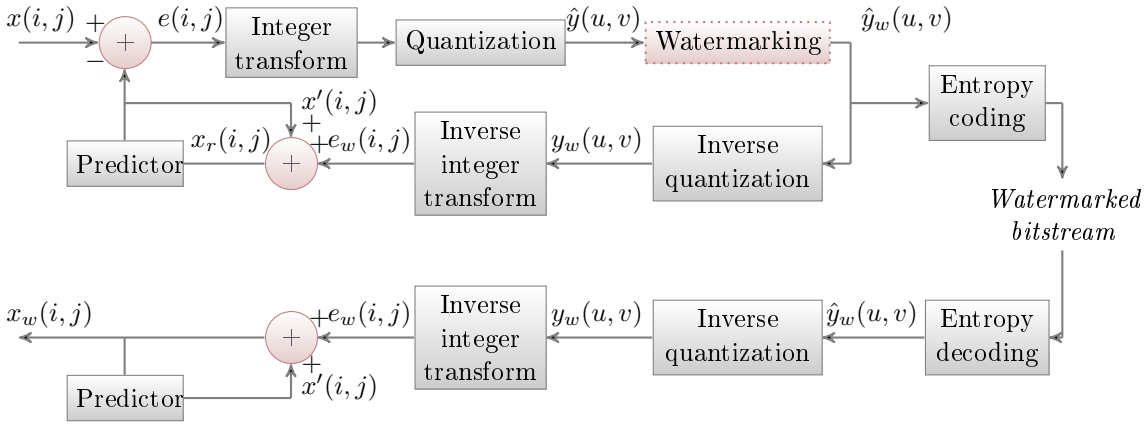


Figure 6.2: The proposed watermarking method inside the reconstruction loop.

6.2.2 Hidden message aware rate distortion

Many encoding parameters like prediction modes, quantization parameter (QP) value and motion vectors are adjusted in the encoding process based on video content and required quality. Since video content is very diverse in nature both spatially and temporally, these parameters vary from scene to scene. Bit allocation algorithms are used to find the best suited values of these parameters to achieve the trade off between bitrate and quality. For RD, Lagrangian bit-allocation is widely used owing to its simplicity and effectiveness. The simplified Lagrangian cost function is $J = D + \lambda R$, where J represents the cost of encoding for a given MB, D is the distortion, λ is the Lagrangian parameter depending on QP value and R is the number of bits to encode a MB. In H.264/AVC, several modes are supported to encode a MB as *intra* or *inter* as shown in Fig. 6.3. For *intra*, H.264/AVC has nine prediction modes for $Intra\_4 \times 4$ block size, while four prediction modes for $Intra\_16 \times 16$ block size. For *inter*, H.264/AVC multiple block size motion estimation starting from 16×16 to 4×4. Cost J for each prediction mode P is calculated and the best mode is selected.

To obtain the cost J for a specific prediction mode P, we first predict the MB for that mode to get residual E. We then apply IT on residual E followed by quantization with some QP value to get QTCs which are then entropy coded. The number of bits R consists of MB header bits and data bits. Then, residual is reconstructed by performing inverse quantization and inverse IT to give the reconstructed residual E'. Generally, the distortion D is either sum of absolute differences (SAD), sum of squared differences (SSD) or sum of absolute transformed differences (SATD) between E and E'. Thus, we end up with the cost J for encoding this MB in the prediction mode P. In a similar fashion, we find cost J for all other prediction modes. The mode which yields the minimum cost is selected as the RD optimized mode for this MB.

Embedding a message in a video bitstream affects quality of the picture. It also

affects the bitrate because this frame is used for prediction after reconstruction. Hence, RD optimization should take into account the embedding of hidden message in QTCs in order to select the best prediction mode. In this case, simplified Lagrangian cost function is $J_w = D_w + \lambda R_w$, for finding the cost for a specific prediction mode. QTCs are first watermarked to get QTC_w which are then entropy coded to find the number of bits R_w to encode MB and reconstructed to measure the distortion D_w. By moving the message embedding process inside the reconstruction loop, it incorporates the best suitable mode for the watermarked blocks.

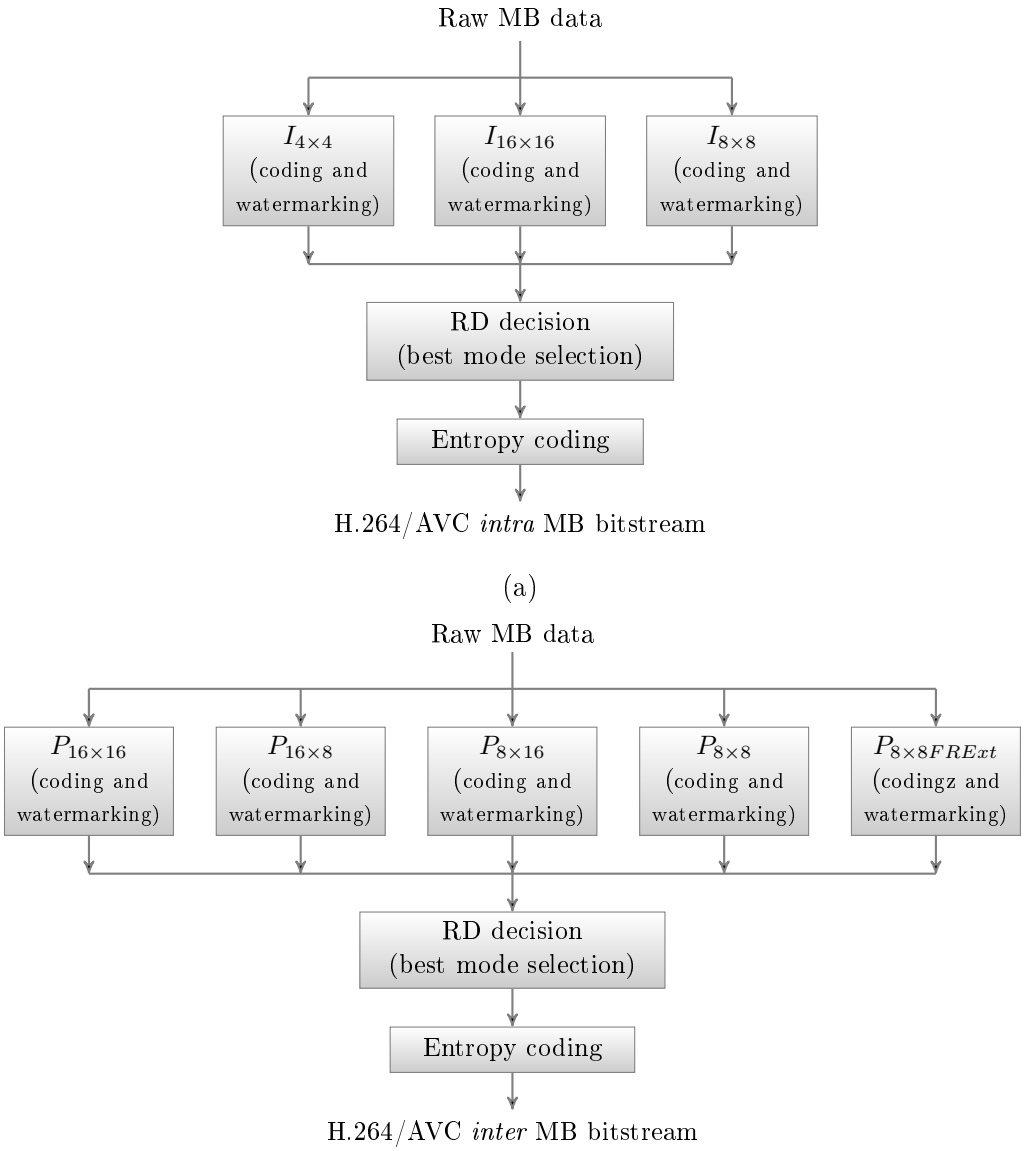


Figure 6.3: Different prediction modes along with watermarking in H.264/AVC for MB type: (a) *Intra* MBs, (b) *Inter* MBs.

In the following sections, we present our work which performs watermark message embedding while taking into account the reconstruction loop.

6.3 Fragile watermarking with high payload

In this section, we propose a high payload fragile watermarking algorithm. The proposed scheme does not target robustness. Rather we have demonstrated the high payload capability of the proposed scheme with controlled increase in bitrate. Hence, the proposed scheme can be used in application where robustness is not required *e.g.*, broadcasting and hiding of meta data. We have used LSB modification approach in the quantized DCT domain. The hidden message is not embedded in all non-zero QTCs, rather we have embedded message in only those QTCs which are above a certain threshold. The threshold value depends upon the number of message bits being embedded. It has two advantages. First, it makes it possible to extract the message on the decoder side. Second, it does not affect the compression efficiency of entropy coding engine to much extent.

In H.264/AVC, *intra* prediction is performed in the spatial domain. So even for *intra* mode, the transform is performed on prediction residuals. In contrast to previous methods which embedded hidden message in DC coefficients, we have embedded message in all the non-zeros AC QTCs having magnitude above a certain threshold. For $Intra\_4 \times 4$ mode, we have not embedded message in DC QTCs, while for $Intra\_16 \times 16$ mode, we have not modified the Hadamard transform coefficients either. It is due to the fact that DC QTCs contain most of the energy and embedding message in them affects the video quality and the bitrate significantly. We have embedded message in LSBs of QTCs keeping in view the following points:

- QTCs which we want to use for watermarking should be non-zero. If a QTC with zero magnitude becomes non-zero in the course of embedding, it will highly affect the compression efficiency of run-length encoding.
- QTC to be used for watermarking should be preferably greater than 1 because there are many QTC with magnitude '1' and in CAVLC, they are also encoded as T1's. Thus changing of number of T1's will affect the compression efficiency of CAVLC.
- Finally, message is embedded in such a fashion that it can be completely extracted on the decoder side.

6.3.1 Message embedding strategy

Integer transform and quantization process for H.264/AVC outputs QTCs, as elaborated in Section 2.3.2. Message embedding process can be performed on quantized transform coefficients (QTCs) as:

$$\hat{y}_w(u,v) = f(\hat{y}(u,v), M, [K]), \tag{6.1}$$

where $f()$ is the watermarking process, M is the hidden message and K is an optional key. $\hat{y}$ is a QTC, while $\hat{y}_w$ is watermarked QTC.

For message embedding in video content, we developed a strategy to embed message in 1 LSB, 2 LSBs or 1 & 2 LSBs together. For n LSB mode, the hidden message is embedded in a QTC in n LSBs if $|QTC| > 2^n - 1$. Owing to this threshold, detection and extraction of message is performed on the decoder side. Algorithm 1 describes the embedding of 1 watermark bit (WMBit) in LSB of $|QTC|$. Here the threshold value is 1. If $|QTC|$ is less than or equal to 1, it will remain unchanged. For $|QTC| \geq 2$, output will be either same or will get modified by ± 1, depending on whether WMBit is '0' or '1'. In this case, we have 0.5 probability that the coefficient will remain unchanged even after being watermarked.

Algorithm 1 The embedding strategy in 1 LSB.

1: **if** $|QTC| > 1$ **then**
2: $\quad |QTC_w| \leftarrow |QTC| - |QTC| \ mod \ 2 + WMBit$
3: **end**

Similarly, Algorithm 2 illustrates the embedding of 2 bits in 2 LSBs of a QTC. By keeping the threshold '3', we can extract the hidden message on decoder side successfully. QTC will remain unchanged if $|QTC| < 4$. If $|QTC| \geq 4$, it will get modified depending on whether WMBits are '00', '01', '10' or '11'. In this case, we have only 0.25 probability that the coefficient will remain unchanged even after being watermarked. So, the compromise in PSNR is of relatively higher value.

Algorithm 2 The embedding strategy in 2 LSBs.

1: **if** $|QTC| > 3$ **then**
2: $\quad |QTC_w| \leftarrow |QTC| - |QTC| \ mod \ 4 + WMBits$
3: **end**

We can also perform a 1 & 2 LSBs embedding together. In this case, we embed message in 0, 1 & 2 LSBs depending on value of $|QTC|$, as shown in Algorithm 3. So we embed 2 WMBits if $|QTC| > 3$, or 1 WMBit if $|QTC| > 1$.

Algorithm 3 The embedding strategy in 1 & 2 LSBs.

1: **if** $|QTC| > 3$ **then**
2: $\quad |QTCw| \leftarrow |QTC| - |QTC| \ mod \ 4 + WMBits$
3: **else**
4: $\quad$ **if** $|QTC| > 1$ **then**
5: $\quad\quad |QTCw| \leftarrow |QTC| - |QTC| \ mod \ 2 + WMBit$
6: **end**

6.3.2 Message extraction strategy

During the extraction process, we can extract the message from watermarked QTCs as:

$$M = g(\hat{y}_w(u,v),[K]), \tag{6.2}$$

where $g()$ is the watermarking detection/extraction process, $\hat{y}_w(u,v)$ is the watermarked QTC, and K is and optional key required for extraction. When using 1 & 2 LSBs watermark extraction, $g()$ can be given as shown in Algorithm 4.

Algorithm 4 The extraction strategy using 1 & 2 LSBs.

1: **if** $|QTC| > 3$ **then**
2: $WMBits \leftarrow |QTC_w| \ mod \ 4$
3: **else**
4: **if** $|QTC| > 1$ **then**
5: $WMBit \leftarrow |QTC_w| \ mod \ 2$
6: **end**

6.3.3 Experimental results

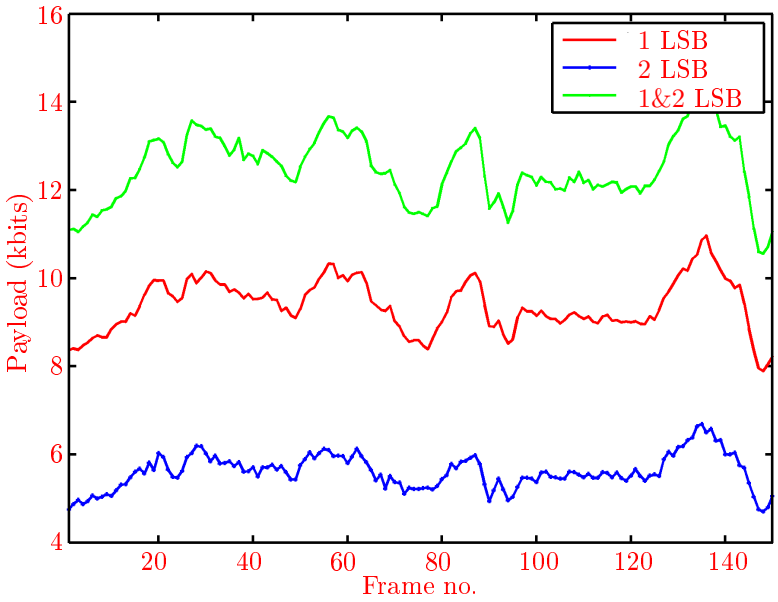
For experimental simulation, we have used the reference implementation of H.264/AVC and benchmark video sequences as explained in Appendix A and applied our method on 150 frames in CIF format. In case of video sequence containing *intra* & *inter* frames, *intra period* has been set to 15 for all the simulations in this section. Hidden message being embedded is generated using pseudo-random number generator.

The detailed results for *intra* frames are explained in Section 6.3.3.1 and Section 6.3.3.2, while Section 6.3.3.3 and Section 6.3.3.4 contain an elaborate discussion for *intra* & *inter* frames. The comparison of our scheme has also been presented with: 1) outside loop embedding, and 2) message embedding in LSBs of all QTCs.

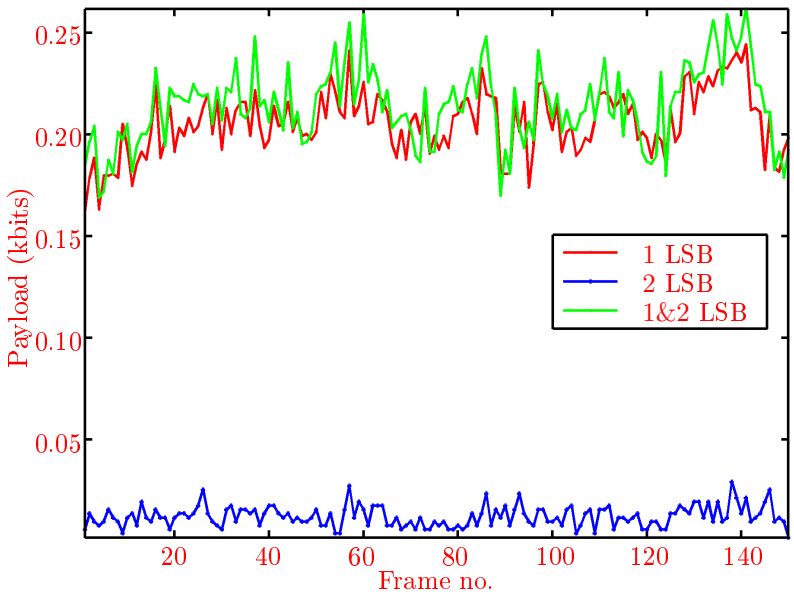
6.3.3.1 Analysis of *intra* frames for bitrate/PSNR/payload trade off

In *intra* frames, non-zero QTCs are present in areas of frames containing texture and edges. These are spatial masking areas and hidden message is naturally embedded in these areas of *intra* frames. To analyze the effect of message embedding on payload, bitrate and PSNR, benchmark sequences have been encoded at QP value 18 for high quality video, and at QP value 36 for low quality video.

Fig. 6.4.a and 6.4.b illustrate the payload for each *intra* frame in *foreman* for QP values 18 and 36 for all of the three watermarking modes namely 1 LSB, 2 LSBs and 1 & 2 LSBs. At QP value 18, we have large number of QTCs which can be watermarked and hence payload is high for all the modes. At QP value 36, we have adequate number of QTCs for 1 LSB mode, so we have enough number of embedded WMBits. But payload for 2 LSBs mode is lower as fewer QTCs have magnitude above threshold for this mode. Fig. 6.5.a and 6.5.b show the effect of message

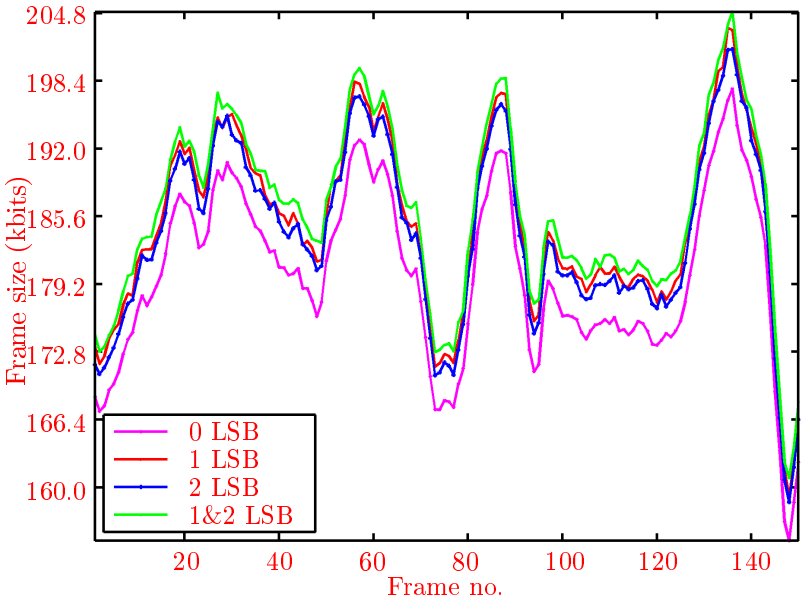


(a)

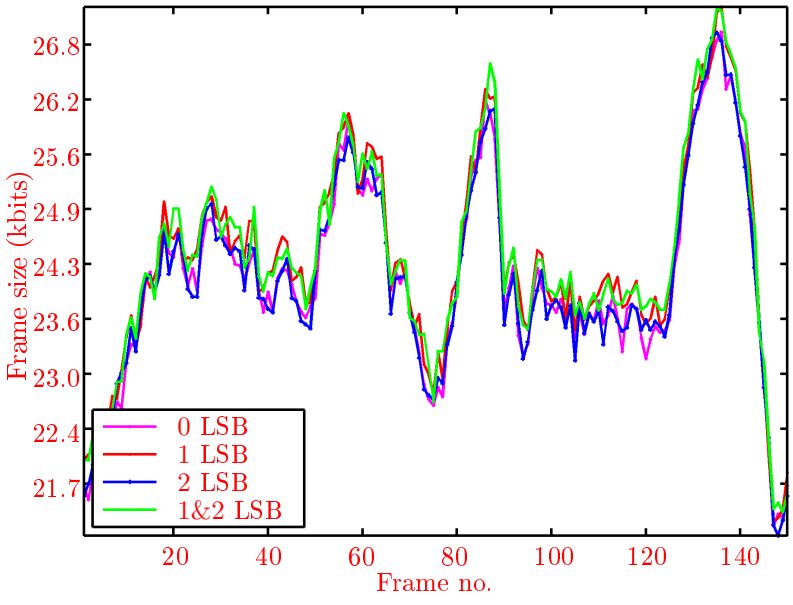


(b)

Figure 6.4: Analysis of payload capability for hidden message embedding of *intra* frames in *foreman* for QP values: (a) 18, (b) 36.

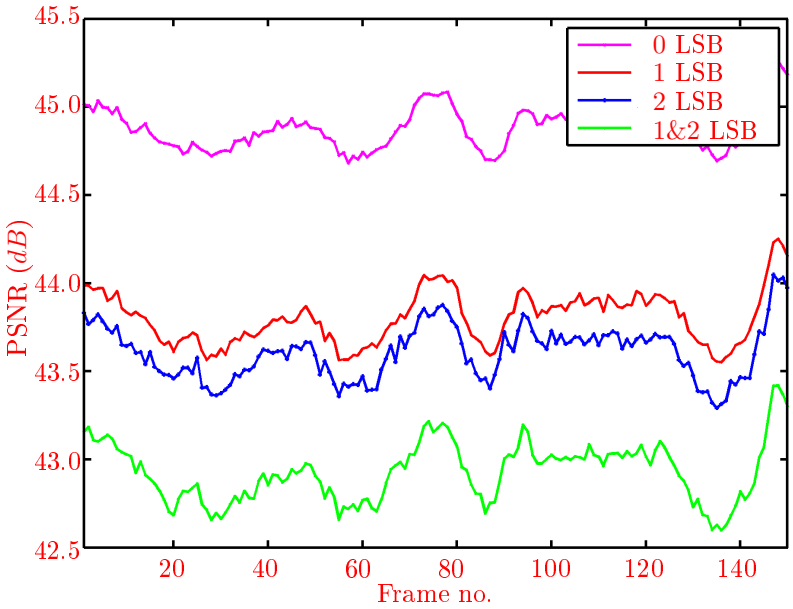


(a)

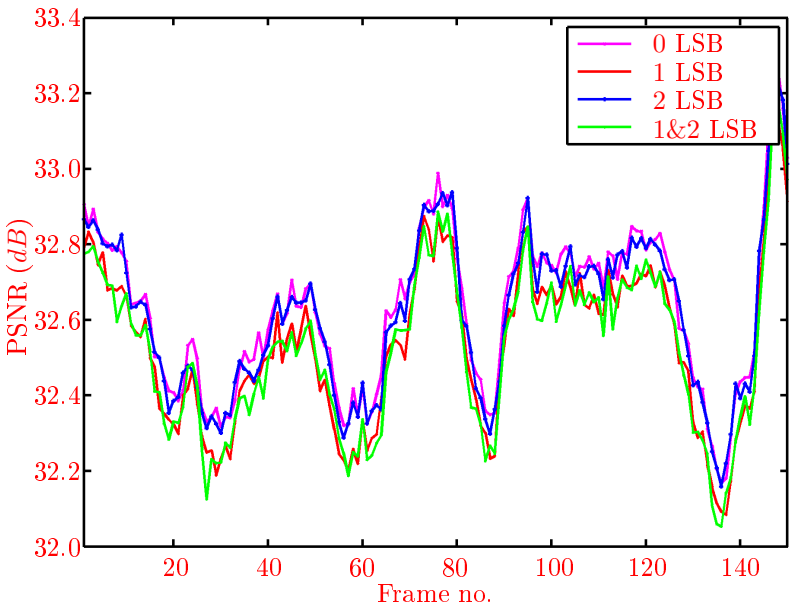


(b)

Figure 6.5: Analysis of bitrate variation for hidden message embedding of *intra* frames in *foreman* for QP values: (a) 18, (b) 36.



(a)



(b)

Figure 6.6: Analysis of change in PSNR for hidden message embedding of *intra* frames in *foreman* for QP values: (a) 18, (b) 36.

embedding on bitrate. Owing to the fact that we have not modified the value of QTCs with magnitude 0 and T1's, bitrate has increased only slightly. This increase in bitrate is due to two reasons. First, watermarked reconstructed QTCs are used for prediction of future MBs which results in more residual and hence increase in bitrate. Second, absolute values of QTCs increase gradually in inverse scan order and entropy coding is designed for this distribution. After WMBit embedding, this order may get disturbed and depends upon the WMBits being embedded. With the embedding of WMBits, QTCs are modified and hence there is a decrease in PSNR as shown in Fig. 6.6.a and 6.6.b. At QP value 18, higher number of coefficients are watermarked and hence a greater reduction in the PSNR is observed. While at QP value 36, we have lesser QTCs to be watermarked, hence less degradation in quality is observed. At QP value 36, few QTCs have magnitude above threshold for 2 LSBs embedding and very few WMBits can be embedded for this mode. But we have adequate number of QTCs with magnitude greater than 1, so we have enough number of WMBits embedded for this mode. Table 6.1 contains payload, bitrate and PSNR analysis for *foreman* and *football* sequences for QP values 18 and 36. For 1 & 2 LSBs mode with QP value 18, the increase in bitrate is 3.34 % and 5.68 %, the payload equals to 312.10 kbps and 396.80 kbps, and the decrease in PSNR is 1.95 dB and 2.03 dB for *foreman* and *football* respectively.

Table 6.1: Results for *intra* for *foreman* and *football* sequences.

QP	Embedding mode (LSBs)	*foreman*			*football*		
		Payload (kbps)	Bitrate (kbps)	PSNR (dB)	Payload (kbps)	Bitrate (kbps)	PSNR (dB)
18	0	0	4504	44.88	0	4730.33	45.32
	1	234.38	4622.4	43.80	293.88	4928.18	44.13
	2	140.13	4600	43.61	188.23	4920.34	43.99
	1& 2	312.10	4654.4	42.93	396.80	4998.85	43.29
36	0	0.00	603.2	32.63	0.00	750.06	32.52
	1	5.15	609.6	32.54	8.26	759.89	32.40
	2	0.30	603.2	32.61	0.84	752.48	32.50
	1& 2	5.35	609.6	32.53	8.73	760.42	32.39

Table 6.2 shows the PSNR, bitrate and payload trade off of our scheme for *intra* sequences of all the benchmark video sequences at QP value 18. We have also compared our scheme with watermark embedding after the reconstruction loop. For our algorithm, decrease in PSNR is 2.65 dB, while it is 19.5 dB for embedding after the encoding loop on average.

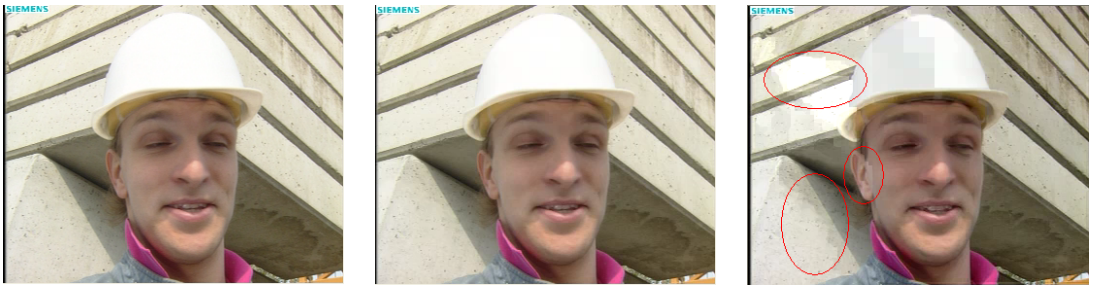
For subjective quality comparison, Fig. 6.7 contains frame #0 of *foreman* and *football* sequences and shows the artifacts in *intra* frames because of message embedding after the encoding loop. Ghost artifacts are encircled red in these video frames. For flat regions, a change in luminance can also be observed in the video frames. These artifacts are because of spatial prediction from top and left blocks.

Table 6.2: Comparison of bitrate, payload and PSNR at QP 18 for watermark embedding inside and outside the reconstruction loop for *intra* sequence with 1 & 2 LSBs mode.

Seq.	Orig.		LSB-inside Loop			LSB-outside Loop		
	PSNR (dB)	Bitrate (mbps)	PSNR (dB)	Bitrate (mbps)	Payload (kbps)	PSNR (dB)	Bitrate (mbps)	Payload (kbps)
Bus	44.52	6.90	40.43	7.32	856.00	24.35	6.93	891.93
City	44.52	6.07	41.66	6.33	611.36	24.91	6.10	640.82
Crew	45.22	4.34	43.80	4.48	290.64	27.72	4.37	305.74
Football	45.32	4.62	43.29	4.88	396.80	27.54	4.64	435.15
Foreman	44.88	4.40	42.93	4.55	312.10	24.54	4.43	315.49
Harbour	44.30	7.20	40.38	7.70	895.74	22.22	7.22	937.46
Ice	47.29	2.25	45.08	2.34	197.28	29.34	2.26	194.80
Mobile	44.59	10.38	41.07	10.63	895.74	22.85	10.38	1640.64
Soccer	45.19	4.42	43.29	4.68	373.39	26.81	4.45	410.98
Avg.	45.09	5.62	42.44	5.88	536.56	25.59	5.64	641.45

Table 6.3: Comparison of bitrate and PSNR of our scheme with watermark embedding in all QTCs for *intra* sequence at QP 18. Reconstruction loop has been taken into account for watermark embedding in all QTCs.

Seq.	Orig.		1 & 2 LSBs mode			1 LSB-all NZs		2 LSBs-all NZs	
	PSNR (dB)	Bitrate (mbps)	PSNR (dB)	Bitrate (mbps)	Payload (kbps)	PSNR (dB)	Bitrate (mbps)	PSNR (dB)	Bitrate (mbps)
Bus	44.52	6.90	40.43	7.32	856.00	38.87	9.55	32.71	12.67
City	44.52	6.07	41.66	6.33	611.36	38.99	8.74	32.74	11.92
Crew	45.22	4.34	43.8	4.48	290.64	39.53	7.61	32.73	11.34
Football	45.32	4.62	43.29	4.88	396.80	39.44	8.01	32.69	11.61
Foreman	44.88	4.40	42.93	4.55	312.10	39.39	7.59	32.76	11.41
Harbour	44.3	7.20	40.38	7.70	895.74	38.68	9.81	32.72	12.94
Ice	47.29	2.25	45.08	2.34	197.28	39.78	6.71	32.53	11.00
Mobile	44.59	10.38	41.07	10.63	895.74	38.63	12.20	32.47	14.31
Soccer	45.19	4.42	43.29	4.68	373.39	39.3	7.85	32.70	11.41
Avg.	45.09	5.62	42.44	5.88	536.56	39.18	8.68	32.67	12.07



(a) original frame (b) Inside loop (c) Outside loop

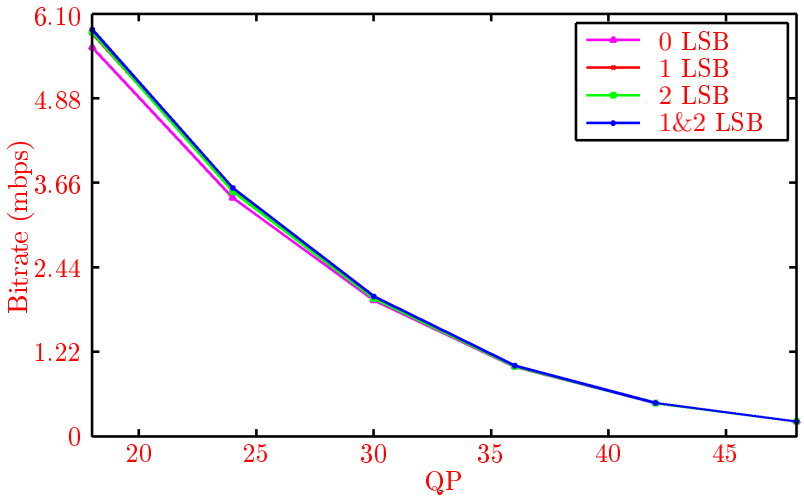
(d) original frame (e) inside loop (f) outside loop

Figure 6.7: Artifacts created in *intra* due to outside loop watermark embedding with 1 & 2 LSBs mode with QP value 18 for frame #0 of *foreman* and *football*.

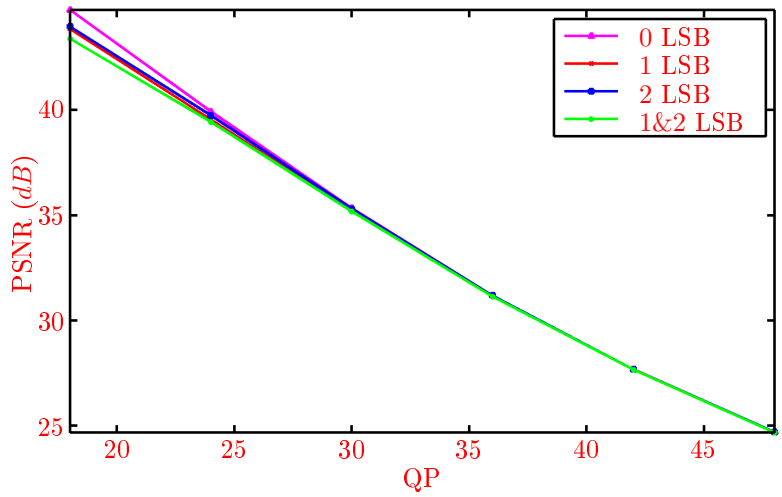
(a) 1 & 2 LSBs (b) 1 LSB-all QTCs (c) 2 LSB-all QTCs

(d) 1 & 2 LSBs (e) 1 LSB-all QTCs (f) 2 LSB-all QTCs

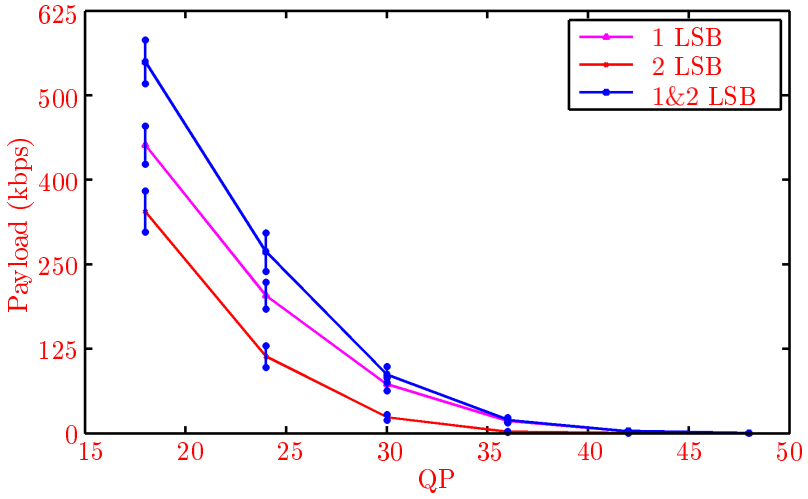
Figure 6.8: Visual comparison of 1 & 2 LSBs mode with naive 1 LSB and 2 LSBs mode (embedding in all QTCs) for *intra* frame #0 for QP value 18.



(a)



(b)



(c)

Figure 6.9: Analysis of watermarking for *intra* frames for benchmark video sequences over the whole range of QP values for: (a) Bitrate, (b) PSNR, (c) Payload. Standard deviation of payload for all the QP values is also shown.

If the encoding loop is not taken into account during the embedding process, a drift or 'increasing error' will be created between the encoder and decoder and different values will be used for prediction on the encoder and decoder side. Hence distortion will increase gradually from top-left corner to bottom-right corner of the image.

To show the efficiency of our embedding scheme, we have compared it with naive watermarking in LSBs of all QTCs, while taking into account the reconstruction loop. Fig. 6.8 shows frame #0 of *foreman* and *football*. In contrast to our algorithm, one can note the noise artifacts in frames in which message embedding is performed in all the QTCs using naive 1 LSB and 2 LSBs embeddings. This noise is owing to the introduction of new frequencies by conversion of zero QTCs to non-zeros. Table 6.3 shows the PSNR, bitrate and payload analysis for naive 1 LSB and 2 LSBs embeddings. We have compared it with 1 & 2 LSBs mode of our algorithm which has the highest trade offs in terms of PSNR and bitrate. When the *skip-mode* is off, payload for CIF resolution at 25 fps will be 3712.5 kbps and 7425 kbps respectively for naive 1 LSB and 2 LSBs embeddings, while payload for our scheme is 536.56 kbps. Table 6.3 shows that average decrease in PSNR for 1 & 2 LSBs mode of our algorithm is 2.65 *dB* while it is 5.91 *dB* for naive 1 LSB embedding and 12.42 *dB* for naive 2 LSBs embedding for all the bench mark sequences. Increase in bitrate is 4.6 % for our algorithm, while it is 54.44 % and 114.76 % for naive 1 LSB and 2 LSBs modifications. Hence trade offs for PSNR and bitrate are so high that naive LSB embeddings cannot be used in practical applications.

6.3.3.2 Analysis of *intra* frames over whole range of QP values

For simulation of our watermarking algorithm over the whole range of QP values, average results for QP values of 18, 24, 30, 36, 42 and 48 have been demonstrated for benchmark video sequences. Fig. 6.9.a illustrates the change in bitrate over the whole range for 1 LSB, 2 LSBs and 1& 2 LSBs. Fig. 6.9.b shows the compromise in PSNR while Fig. 6.9.c illustrates the payload capability at different QP values. PSNR graph is linear and from QP 36 onward, there is no considerable degradation of quality. It is important to note that it is only the PSNR which has a linear graph, while, bitrate and payload graphs are not linear. The reason is that quantization value is exponential function of QP value with step 6 and PSNR is a logarithmic measure.

6.3.3.3 Analysis of *inter* frames for bitrate/PSNR/payload trade off

Inter frames contain both *intra* and *inter* MBs. Prediction is performed from previous video frames in case of *inter* MBs, and from top and left blocks in case of *intra* MBs. Non-zero QTCs are found in the parts of frames containing motion and texture. Hidden message is naturally embedded in those areas as these are temporal masking areas in *inter* frames. In a video sequence, after every *intra* frame, first few *inter* frames are better predicted and contains lesser residual errors. Hence message embedding affects quality and compression ratio. But as we go away from *intra*

frames, accumulated errors appear and message embedding does not affect much the quality of *inter* frames. On average, after 5 *inter* frames followed by *intra* frame, the ratio of payload to frame size of *inter* frames is comparable to that of *intra* frames especially at lower QP values.

Table 6.4: watermarking results with *intra* and *inter* frames of *foreman* and *football* video sequences for QP value of 18.

QP	Embedding mode (LSBs)	*foreman*			*football*		
		Payload (kbps)	Bitrate (kbps)	PSNR (dB)	Payload (kbps)	Bitrate (kbps)	PSNR (dB)
I	0	0	4508.80	44.88	0	4790.88	45.29
	1	233.80	4627.20	43.80	295.62	4979.20	44.10
	2	139.65	4604.80	43.61	196.25	4974.40	43.90
	1& 2	311.30	4660.80	42.92	402.42	5049.76	43.20
P	0	0.00	2107.20	44.54	0	3558.72	44.76
	1	34.45	2148.80	44.30	172.91	3673.76	43.65
	2	7.00	2132.80	44.45	82.01	3731.36	43.68
	1& 2	38.25	2166.40	44.23	207.43	3798.24	43.05
I+P	0	0	2267.20	44.56	0	3640.80	44.79
	1	47.73	2313.60	44.27	181.09	3760.64	43.69
	2	15.83	2297.60	44.39	89.62	3814.24	43.70
	1& 2	56.45	2332.80	44.14	220.43	3881.76	43.06

Table 6.5: watermarking results with *intra* and *inter* frames of *foreman* and *football* video sequences for QP value of 36.

QP	watermarking mode (LSBs)	*foreman*			*football*		
		Payload (kbps)	Bitrate (kbps)	PSNR (dB)	Payload (kbps)	Bitrate (kbps)	PSNR (dB)
I	0	0	601.60	32.6128	0	781.28	32.42
	1	4.96	608.48	32.5229	9.08	791.20	32.30
	2	0.28	601.44	32.5894	0.97	782.40	32.38
	1& 2	5.19	609.44	32.5202	9.70	792.64	32.28
P	0	0	117.76	32.3527	0	441.92	31.62
	1	0.11	117.92	32.3148	2.91	448.00	31.55
	2	3.1×10^{-3}	117.12	32.3356	0.12	443.04	31.62
	1& 2	0.13	118.72	32.3177	2.93	448.16	31.55
I+P	0	0	149.92	32.3701	0	464.48	31.67
	1	0.44	150.56	32.3287	3.32	470.88	31.60
	2	0.02	149.44	32.3526	0.18	465.60	31.67
	1& 2	0.47	151.36	32.3312	3.38	471.04	31.60

Fig. 6.10, Fig. 6.11 and Fig. 6.12 have been used for payload, bitrate and PSNR

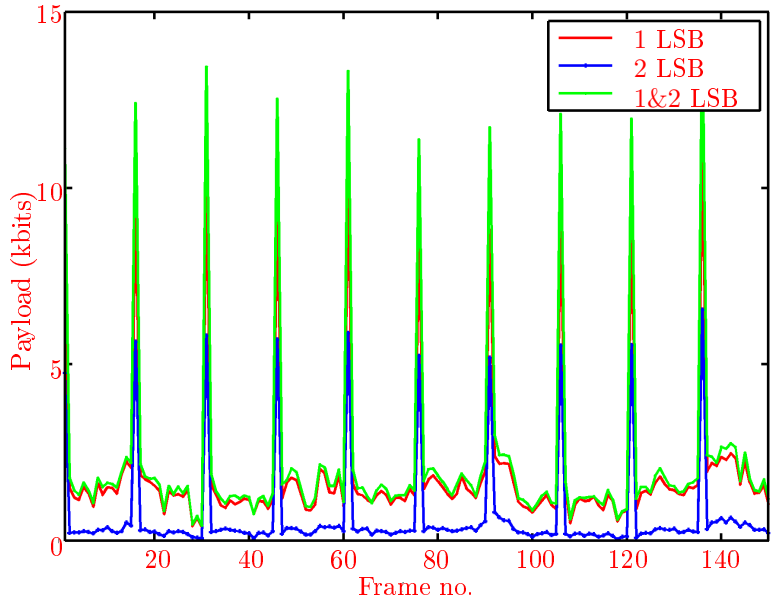
analysis respectively at QP values of 18 and 36 for *foreman* sequence. One can note that payload is adequate at QP value of 18 and it decreases with increase in value of QP. In fact, at QP value of 36, we have very few QTCs with magnitude above threshold for 1 LSB mode, let alone the 2 LSBs mode. Table 6.4 and Table 6.5 show the payload, bitrate and PSNR analysis for *foreman* and *football* sequences at QP value of 18 and 36 for *intra* & *inter* frames. For *foreman* sequence with 1 & 2 LSBs mode at QP value of 18, the increase in bitrate is 2.81 % and 2.89 %, the payload is the 38.25 kbps and 56.45 kbps and PSNR decrease is 0.22 *dB* and 0.20 *dB* for *inter* and *intra* & *inter* respectively. While for *football* sequence, the increase in bitrate is 6.73 % and 6.62 %, the payload is the 207.43 kbps and 220.43 kbps and PSNR decrease is 1.71 *dB* and 1.73 *dB* for *inter* and *intra* & *inter* respectively. *Football* sequence has greater payload capacity than *foreman* especially in P frames. It is because of texture and high motion in *football* sequence.

Overall analysis for all the benchmark video sequences is given in Table 6.6. It also contains comparison with message embedding after the encoding loop. Decrease in PSNR for our scheme is 1.38 *dB*, while it is 19.2 *dB* for embedding after the encoding loop. For subjective quality comparison with outside loop watermark embedding, Fig. 6.13 contains frame #89 of *foreman* and *football* sequences and shows the artifacts in *inter* frame because of watermark embedding after the encoding loop. One can note that *inter* frame is heavily distorted in this case. The result shows that watermark embedding after the encoding loop distorts the video frame in case of *inter*. Hence, message embedding after the encoding loop is not a workable solution for state of the art video codecs because of spatial and temporal prediction.

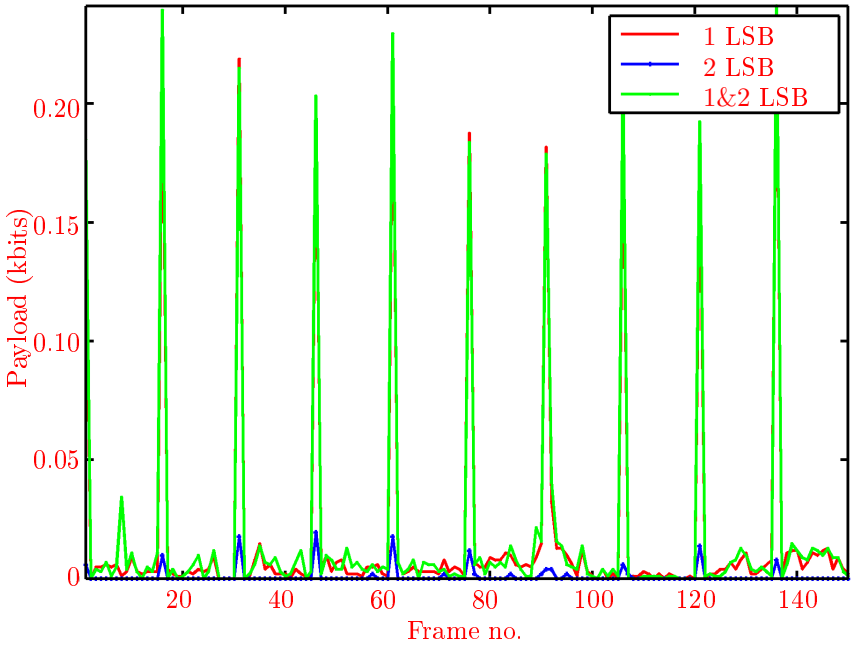
For comparison, with naive message embedding in LSBs of all QTCs, Fig. 6.14 shows the frame #28 of *foreman* and *football*. Just like *intra* frames, one can easily note the noise artifacts in frames in which message embedding is performed in all the QTCs using 1 LSB and 2 LSBs embeddings, which is owing to the introduction of new frequencies. Table 6.7 shows the PSNR, bitrate and payload analysis at QP value of 18. Payload, for naive 1 LSB and 2 LSBs embeddings, will be similar to *intra* frames *i.e.* 3712.5 kbps and 7425 kbps respectively. Average decrease in PSNR for our algorithm is 1.38 *dB* while it is 6.62 *dB* for naive 1 LSB embedding and 12.93 *dB* for naive 2 LSBs embedding. Increase in bitrate is 4.6 % for our algorithm, while it is 101.23 % and 216.09 % for naive 1 LSB and 2 LSB modifications. Such high trade offs for PSNR and bitrate make it inappropriate for practical applications.

6.3.3.4 Analysis of *inter* frames over whole range of QP values

The overall performance analysis of benchmark video sequences with *intra period* 15 is shown in the form of graphs in Fig. 6.15. Fig. 6.15.a illustrates the effect of watermarking on bitrate while Fig. 6.15.b shows the change in PSNR for different message embedding modes. Fig. 6.15.c illustrates the payload capability of our algorithm along with standard deviation of payload at different QP values. Between 1 LSB and 2 LSBs embedding modes, 1 LSB performs better having higher payloads

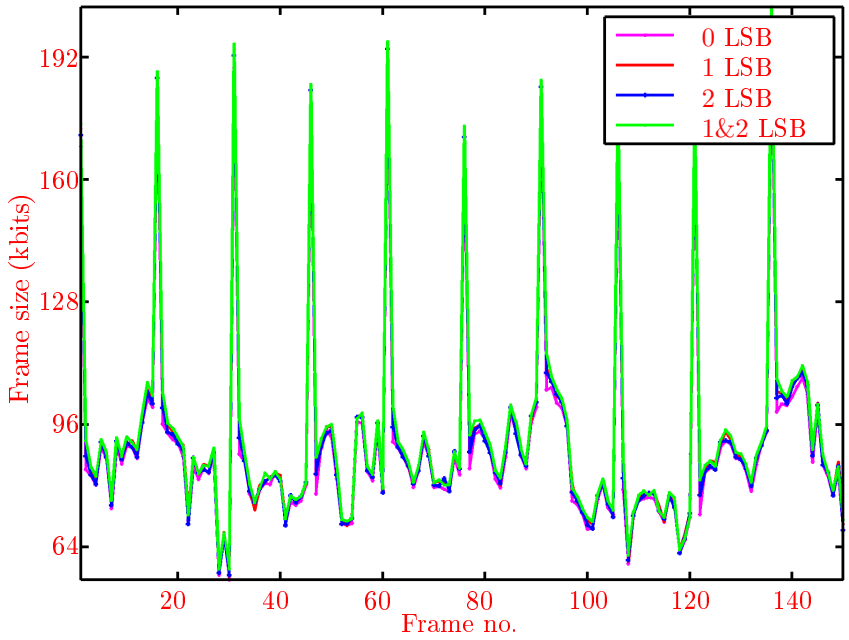


(a)

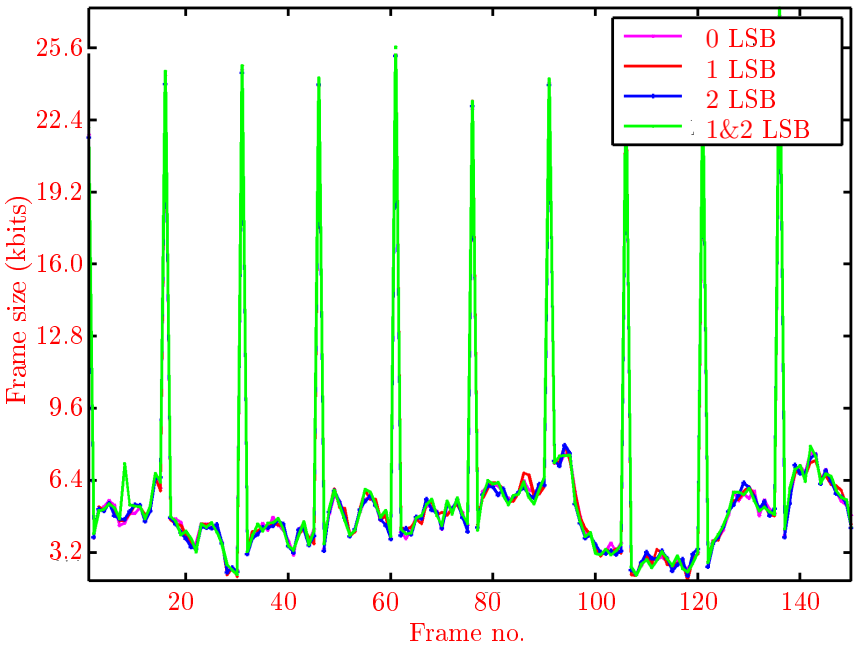


(b)

Figure 6.10: Analysis of payload capability for message embedding of *intra* & *inter* frames for *foreman* for QP values: (a) 18, (b) 36.

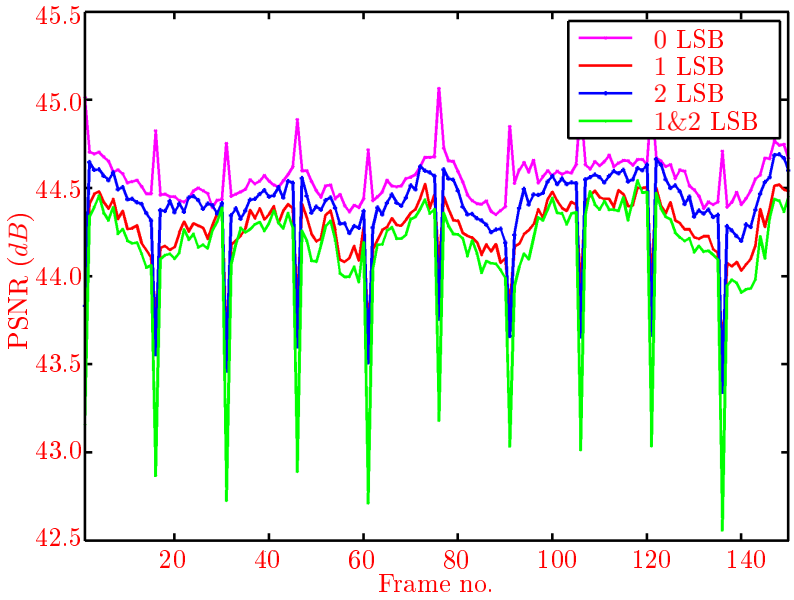


(a)

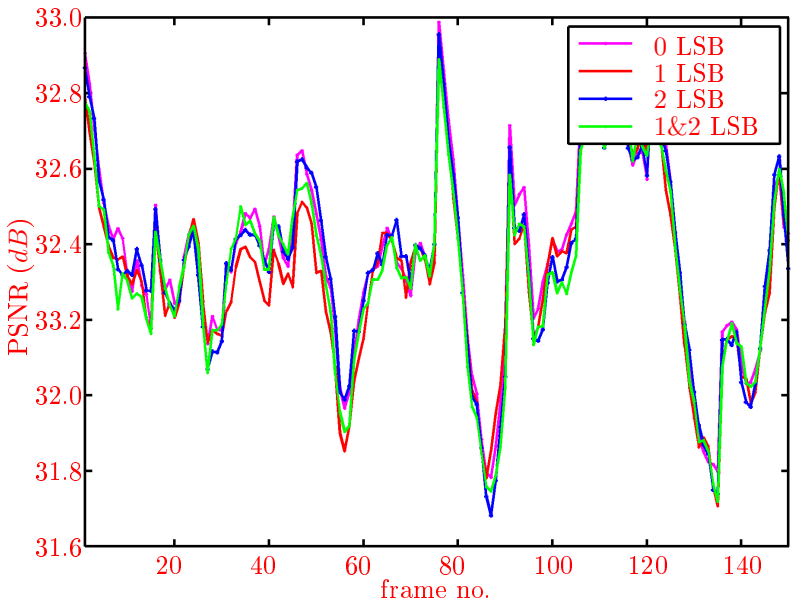


(b)

Figure 6.11: Analysis of change in bitrate for message embedding of *intra* & *inter* frames for *foreman* for QP values: (a) 18, (b) 36.



(a)



(b)

Figure 6.12: Analysis of change in PSNR for message embedding of *intra* & *inter* frames for *foreman* for QP values: (a) 18, (b) 36.

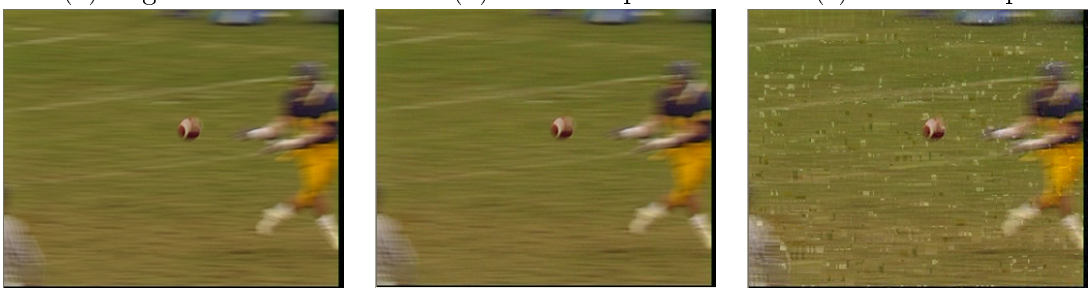
Table 6.6: Comparison of bitrate, payload and PSNR for message embedding inside and outside the reconstruction loop for *intra* & *inter* sequence with 1 & 2 LSBs mode. QP value is 18 and *intra period* is 15.

Seq.	Orig.		LSB-inside Loop			LSB-outside Loop		
	PSNR (dB)	Bitrate (mbps)	PSNR (dB)	Bitrate (mbps)	Payload (kbps)	PSNR (dB)	Bitrate (mbps)	Payload (kbps)
Bus	44.27	4.00	42.21	4.22	278.38	24.12	4.34	356.86
City	44.36	2.57	43.80	2.66	87.14	25.16	2.76	115.12
Crew	44.80	3.12	44.00	3.24	111.90	28.48	3.50	164.94
Football	44.79	3.56	43.06	3.79	220.43	26.94	3.99	320.76
Foreman	44.56	2.22	44.14	2.28	56.46	25.04	2.53	92.18
Harbour	44.18	4.45	42.05	4.72	330.15	22.46	4.70	380.64
Ice	46.93	1.06	46.27	1.11	46.03	29.14	1.29	89.51
Mobile	44.18	5.70	41.00	5.96	526.03	22.97	5.98	581.07
Soccer	44.82	2.35	43.99	2.46	100.12	25.86	2.64	153.12
Avg.	44.77	3.23	43.39	3462.43	195.18	25.57	3.52	250.47

Table 6.7: Comparison of bitrate and PSNR of our scheme with message embedding in all QTCs for *intra* & *inter* sequence with QP 18.

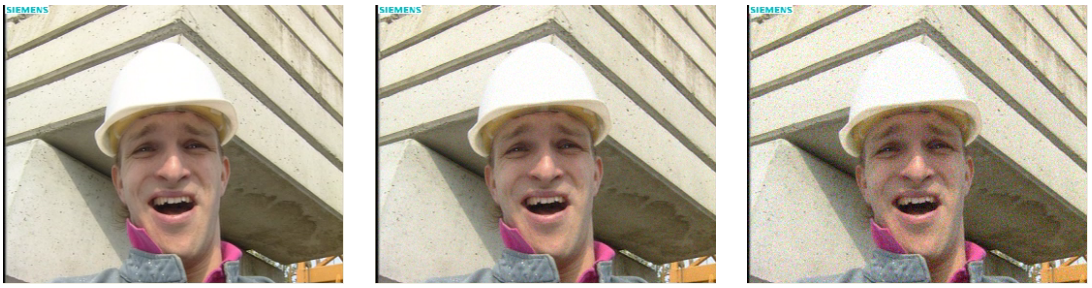
Seq.	Orig.		1 & 2 LSBs mode			1 LSB-all NZs		2 LSBs-all NZs	
	PSNR (dB)	Bitrate (mbps)	PSNR (dB)	Bitrate (mbps)	Payload (kbps)	PSNR (dB)	Bitrate (mbps)	PSNR (dB)	Bitrate (mbps)
Bus	44.27	4.00	42.21	4.22	278.38	37.86	6.87	31.79	10.36
City	44.36	2.57	43.80	2.66	87.14	38.28	5.82	32.07	9.82
Crew	44.80	3.12	44.00	3.24	111.90	38.33	6.36	31.97	10.16
Football	44.79	3.56	43.06	3.79	220.43	38.16	6.87	31.77	10.54
Foreman	44.56	2.22	44.14	2.28	56.46	38.45	5.76	32.11	9.85
Harbour	44.18	4.45	42.05	4.72	330.15	37.65	7.02	31.79	10.26
Ice	46.93	1.06	46.27	1.11	46.03	38.70	6.08	31.51	10.31
Mobile	44.18	5.70	41.00	5.96	526.03	37.50	7.74	31.60	10.57
Soccer	44.82	2.35	43.99	2.46	100.12	38.41	5.98	31.95	10.04
Avg.	44.77	3.23	43.39	3.38	195.18	38.15	6.50	31.84	10.21

(a) original frame (b) Inside loop (c) Outside loop



(d) original frame (e) inside loop (f) outside loop

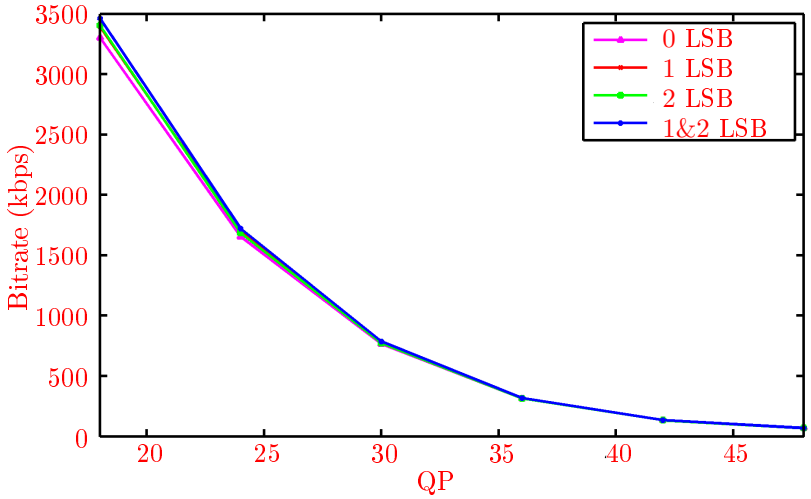
Figure 6.13: Artifacts created in *inter* frames due to outside loop message embedding with 1 & 2 LSBs mode with QP 18 for frame #89 of *foreman* and *football*.



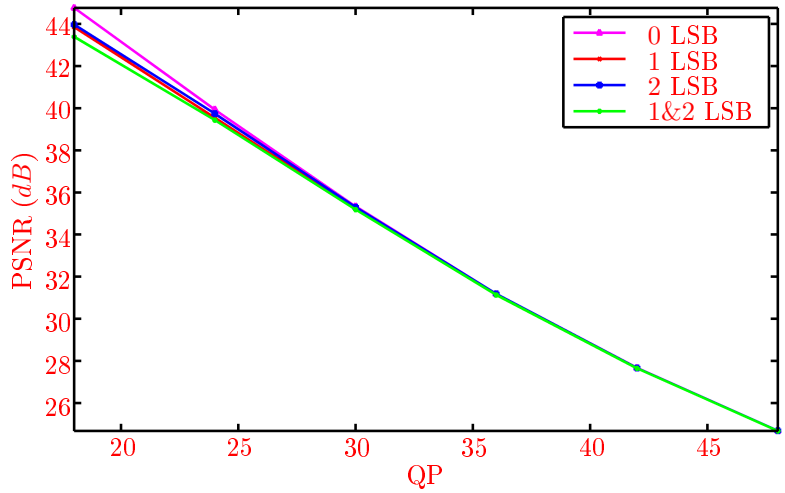
(a) 1 & 2 LSBs (b) 1 LSB-all QTCs (c) 2 LSB-all QTCs

(d) 1 & 2 LSBs (e) 1 LSB-all QTCs (f) 2 LSB-all QTCs

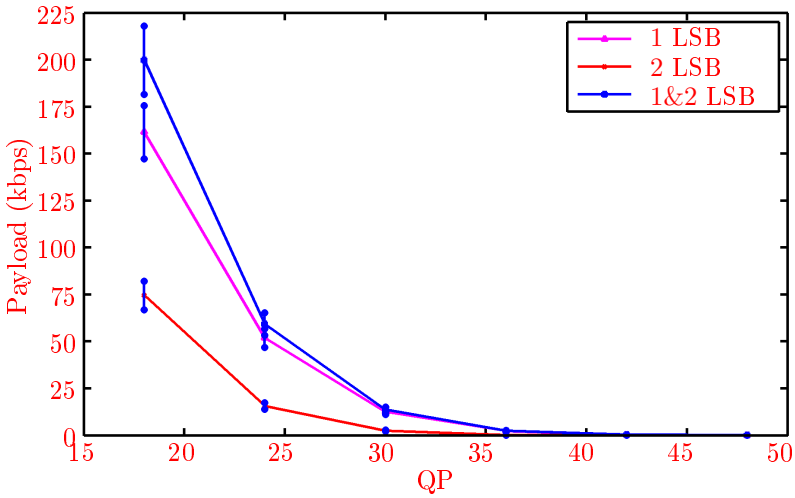
Figure 6.14: Visual comparison of 1 & 2 LSBs mode with naive 1 LSB and 2 LSBs mode (embedding in all QTCs) for *inter* frame #28 for QP value 18.



(a)



(b)



(c)

Figure 6.15: Analysis of watermarking for *intra* & *inter* frames for benchmark video sequences over the whole range of QP values for: (a) Bitrate, (b) PSNR, (c) Payload along with its standard deviation for all the QP values.

and minimum increase in bitrate. In 2 LSBs mode, 2 bits are embedded in the same coefficient and thus magnitude of compromise is higher. 2 LSBs should be used in combination with 1 LSB mode (1 & 2 LSBs embedding mode) when higher payload is required. One can note that just like bitrate, payload varies with QP both in case of *intra* and *inter*.

For the sake of comparison, let us define the *watermark cost* be the increase in bitrate (in bits) per watermark bit. In case of *intra*, the *watermark cost* is 0.06 and 0.15 at QP values 18 and 36 respectively. These results are well better than 1.54, the result presented in [Noorkami 2008]. In case of *intra* and *inter*, we get *watermark cost* of 0.15 and 0.42 at QP values 18 and 36 respectively which is well better than 1.50, the result of work presented in [Noorkami 2008]. At higher QP values, we do not have lot of non-zero QTCs which can be watermarked but the ratio between payload and bitrate is still conserved.

6.3.4 Synopsis

We have designed and analyzed a new video watermarking scheme with high payload for H.264/AVC. Our scheme embeds RD optimized hidden message in QTCs for both *intra* and *inter* frames. Owing to taking into account the reconstruction loop, the proposed scheme offers consistent payload capability to H.264/AVC standard at different bitrates without adversely affecting the overall bitrate and PSNR of the video bitstream. Hidden message is embedded in those regions of *intra* frames which contain edges and texture and for *inter* frames, message embedding is naturally done in temporal masking regions, which contains motion and texture. Experimental results have demonstrated that *inter* frames can be really used for message embedding owing to its motion and texture masking.

Along with fragile watermarking scheme for H.264/AVC, robust watermarking schemes are equally important and have applications in many critical fields. Since multimedia content is increasing day by day, one of the most important applications of robust watermarking is active fingerprinting to trace the colluders in case of illegal distribution of video content. In the following Section, we present a scheme for active fingerprinting of H.264/AVC video.

6.4 Fingerprinting of H.264/AVC for traitor tracing

Active fingerprinting is used for traitor tracing in case of illegal distribution of video content, as described in Chapter 5. It has lot of importance for surveillance as well as commercial video content. In this Section, we propose a scheme for active fingerprinting of H.264/AVC video. It works in the following steps:

- Separate Tardos fingerprinting code is generated for each user.
- H.264/AVC video is watermarked off-line in 2^q versions containing different symbols, where q are the bits being embedded in one independent coded unit

(called *slice*). In *intra* frame, we encode a single frame in more than one *slices* to encode more than 1 bits in it.

- The online content server just provides the right *slices* according to the user fingerprint code.
- On the decoding side, fingerprint is first extracted, followed by the accusation process accusing some users (or nobody) based on this extracted sequence.

Spread spectrum watermarking for H.264/AVC is explained in Section 6.4.1, while embedding strategy for fingerprinting code is detailed in Section 6.4.2. It is followed by experimental simulation in Section 6.4.3, wherein collusion attacks have been performed on fingerprinted video in pixel domain to prove the robustness of the proposed scheme against different linear and non-linear collusion attacks.

6.4.1 Spread spectrum strategy

Many robust watermarking techniques exist in literature. Spread spectrum watermarking technique has been selected for our investigation because it offers certain robustness [Cox 1997] and capacity [Moulin 2003]. Spread spectrum is resistant against number of non-malicious attacks, where the watermarks have a component-wise Gaussian distribution and are statistically independent. It was argued to be highly resistant to non-malicious collusion attacks [Hartung 1999]. Spread spectrum embeds the watermark in overlapped regions and this spreading makes it challenging to change even a single bit at will. This confines the effect of a non-malicious colluder's action to a milder form of collusion from the designer's point of view. Let $S(i,j)$ be the i^{th} bit of Tardos fingerprinting code of user j which is to be embedded into a block of host vector X. To increase the energy of the embedding bit, we specify a scaling parameter α, which is decided based on the human perception. So the watermarked block is given as:

$$Y = X + \alpha U_i(-1)^{S(i,j)}, \tag{6.3}$$

with $S(i,j) = 0$ or 1 and U_i is a Gaussian sequence of size l. The attacked watermarked signal is $Z = Y + n$, where n is the noise due to attack. The watermark bit $\tilde{S}(i,j)$ is extracted from Z by the linear correlation of Z and U_i of length l as:

$$\tilde{S}(i,j) = \begin{cases} 0, & \text{if } \sum_{j=0}^{l} Z[j]U_i[j] > 0 \\ 1, & \text{if } \sum_{j=0}^{l} Z[j]U_i[j] < 0. \end{cases} \tag{6.4}$$

6.4.2 Embedding strategy for fingerprinting code

For embedding a Tardos code in QTCs of H.264/AVC, embedding should be done inside the reconstruction loop, since embedding fingerprinting code after reconstruction loop creates two problems. First, we start reconstruction on the encoder side with QTCs while on the decoder side we start decoding with watermarked QTCs.

This results in a mismatch on the decoder side, which keeps on increasing because of the prediction process of video codec. Because of this mismatch, the difference in PSNR is very significant even for *intra* frames. Second, Rate Distortion (RD) bit allocation algorithm works in quantization module and any change in bitrate/quality trade off because of the watermarking of QTCs is not taken into account.

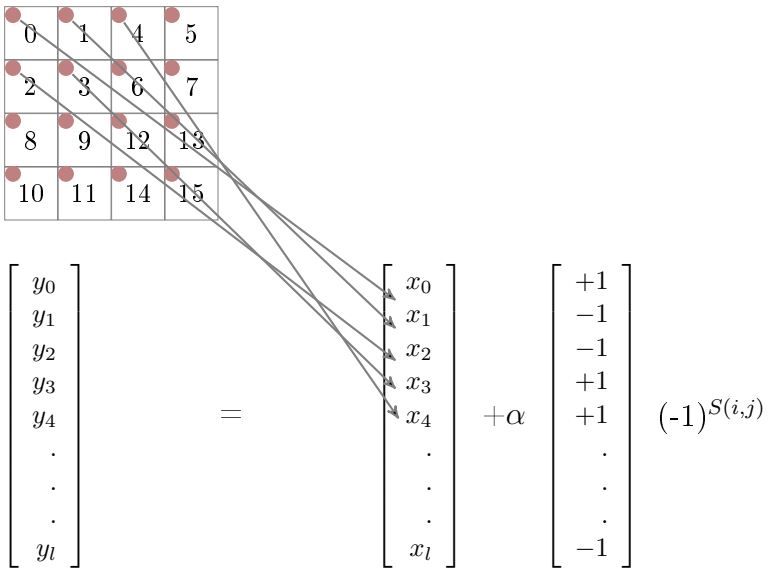


Figure 6.16: Spread spectrum embedding of $S(i,j)$ Tardos code bit in copy of user j.

To solve both the problems, watermark embedding should be performed inside the reconstruction loop as shown in Fig. 6.2. For embedding of m bits of Tardos code S_j in the personal copy of user j, the content is divided into m blocks and 1 bit is embedded into each block. In $Intra\_4 \times 4$ mode, scanning of 4×4 of H.264/AVC, The scanning of these 4×4 blocks inside MB is not in a raster scan fashion. So while we will create a vector X for spread spectrum while taking this scan into account. In our case, X consists of DC transform coefficients of 4×4 IT blocks. Each bit of Tardos code of each use j is embedded into the host vector X using spread spectrum insertion as illustrated in Fig. 6.16. For embedding of m bits of Tardos code S_j in the personal copy of user j, the content is divided into m blocks and 1 bit is embedded into each block. In *intra* 4×4 mode, scanning of 4×4 of H.264/AVC, The scanning of these 4×4 blocks inside MB is not in a raster scan fashion. So while we will create a vector X for spread spectrum while taking this scan into account. In our case, X consists of DC transform coefficients of 4×4 IT blocks. Each bit of Tardos code of each use j is embedded into the host vector X using spread spectrum insertion as illustrated in Fig. 6.16.

6.4.3 Experimental simulation

We have used the benchmark video sequences in CIF resolution for simulation analysis. We have selected *Intra*_4 × 4 MB mode for encoding *intra* sequence along with CAVLC entropy coding scheme. For generation of binary Tardos code, the parameter values are: $n = 100$, $\varepsilon_1 = 10^{-3}$, $c_0 = 20$ and $m = 92104$. Normally, in traitor tracing, $\varepsilon_1 = 10^{-6}$, we have selected $\varepsilon_1 = 10^{-3}$ to reduce the code-length and hence, the simulation time. 10 bits of Tardos code were embedded per frame using robust spread spectrum watermarking. Hence it takes 9211 frames of CIF resolution to embed the code, which is about 6 minutes of video at 25fps. Nine benchmark video sequences having total frames 2450, as explained in Appendix A, have been concatenated in a repeated fashion to encode the desired number of frames.

For accusation process, an accusation sum A_j is calculated for each user j as explained in Chapter 5. A_j for accused users is with a Gaussian centered at $\mu = \frac{2m}{c\pi}$, while A_j for innocent users is modeled with a Gaussian centered at 0. Accused users (the traitors) have a score above $\mu - \sqrt{m}$ (*i.e.* $\frac{2m}{c\pi} - \sqrt{m}$), where $\sqrt{m}$ is the standard deviation of the Gaussian.

In this work, linear and non-linear collusion attacks have been performed in the spatial domain. Linear attacks include averaging and copy-and-paste attack. While typical nonlinear collusion attacks are minimum/maximum/median attacks, minmax attack and modified negative attack.

Fig. 6.17 illustrates the PSNR of the pirated video which has been attacked using linear and non-linear attacks. It also shows the PSNR of original copy. It is evident that in case of average/median/minmax attacks, the video quality gets better with increase in number of colluders. While for minimum/maximum attack, the quality of video slightly decreases as the number of colluders increases.

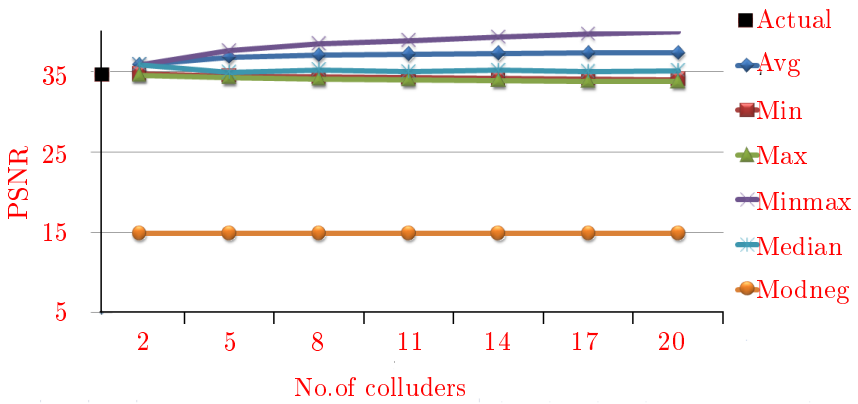


Figure 6.17: PSNR of colluded video content, which has been generated using collusion attacks.

Table 6.8 shows the number of colluders which have been successfully traced by analyzing a pirated video content. In most of the cases, the colluders have been successfully traced. The attack which makes the video non-traceable also degrades

Table 6.8: Number of colluders traced from the K-colluded copy, generated using different collusion attacks.

	Number of colluders detected for attacks					
K	avg	min	max	median	minmax	modNeg
2	2	2	2	2	2	2
5	5	5	5	5	5	5
8	8	8	8	8	8	6
11	11	10	10	10	10	7
14	14	13	13	13	13	9
17	16	15	16	16	16	10
20	18	18	18	19	18	11

the quality of the video very badly. For example, in case of modified negative attack, we could detect only few of the colluders, but PSNR of the attacked video is 15 dB. Averaging attack does not degrade the visual quality rather PSNR gets better in some cases but we can also detect the colluders with very high confidence value. Among the non-linear attacks, modified negative attack makes the accusation process difficult but it severely degrades the video quality.

6.4.4 Synopsis

In this Section, we have demonstrated the fingerprinting of H.264/AVC using Tardos Fingerprinting codes. Spread spectrum robust watermarking technique has been used for embedding of fingerprinting code in personal copies. Our experimental analysis verifies that embedding of Tardos code using spread spectrum watermarking technique is a very good design indeed. The colluded video content can be successfully analyzed to trace the colluders until its quality becomes unacceptable. It is important to note that for large multimedia content, we can have slices of big size and hence the embedding will be more robust.

In this Section, collusion attacks have been performed in spatial domain. This work can be extended to have stronger collusion attacks inside H.264/AVC taking into account different parameter like motion vectors, prediction modes etc. Moreover, the fingerprint embedding technique can be improved by: 1) incorporating perceptual masking, 2) using informed robust watermarking technique.

6.5 Summary

In this chapter, we have presented an efficient method to embed message in *intra* & *inter* frames, during compression step of H.264/AVC video. The fragile watermarking method has the peculiarity of having high payload with controlled increase in payload and little compromise on PSNR value, simultaneously [Shahid 2009b]. High payload of embedding may somehow go against the robustness but the latter has

Figure 6.18: Colluded fingerprinted video sequences of *bus, city, foreman and football*: (a) Original frame, (b) Fingerprinted frame, (c) Average Collusion (17 colluders), (d) Modified negative collusion (8 colluders), ghost artifacts are prominent in video attacked by modified negative collusion with PSNR around 15 dB.

never been the dominant goal. With the spread spectrum spreading of watermark message while taking into account the reconstruction has proved to be robust. The utilization of robust watermarking for embedding fingerprinting code has promised very good results [Shahid 2010d]. It can detect several pirates even in case of modified negative collusion attacks, when the PSNR of the video comes down to 15 dB along with ghost artifacts which are created in the video because of this collusion attack. The utilization of robust watermarking for embedding fingerprinting code has promised very good results [Shahid 2010d]. It can detect several pirates even in case of modified negative collusion attacks, when the PSNR of the video comes down to 15 dB along with ghost artifacts which are created in the video because of this collusion attack.

Chapter 7

Selective Encryption of H.264/AVC and AVS Video

Contents

7.1 Introduction

Le cryptage est utilisé pour la confidentialité et l'accès limité aux utilisateurs autorisés. Une introduction sur le cryptage et des travaux récents dans le domaine de la protection d'images et de vidéos a été présentée dans le Chapitre 4. Le cryptage complet fournit la sécurité la plus haute, mais n'est pas viable à cause de la taille importante des contenus multimédia. De plus, nous n'avons pas toujours besoin d'une sécurité complète pour protéger la valeur commerciale de contenus multimédia. Le chiffrage sélectif est alors utilisé pour la protection vidéo, où la qualité visuelle de la vidéo cryptée est dégradée. Le problème du cryptage sélectif est appliqué à la compression de vidéos au format H.264/AVC et AVS. Nous avons transformé le module de codage entropique en un module de crypto-compression

et qui exécute le chiffrage et la compression en même temps. Des contraintes de temps réel ont été respectées en conservant le débit du format original. L'algorithme proposé exige une puissance de traitement minimale.

Encryption is used for confidentiality and restricted access only to authorized users. Basic introduction of encryption and recent work in this domain for protection of image and video protection is presented in Chapter 4. Full encryption provides the highest security, but is not viable because of huge size of multimedia content. Moreover, we do not need the full security to protect the commercial value of multimedia content. Selective encryption is hence used for video protection, wherein the visual quality of the encrypted video is degraded. The problem of selective encryption (SE) is addressed along with the compression for H.264/AVC and AVS. We have transformed the entropy coding module into a crypto-compression module and it performs encryption and compression at the same time. Real-time constraints have been met by having the bitrate unchanged and keeping the bitstream format complaint. The proposed algorithm requires minimal processing power.

This chapter is organized as follows. Section 7.2 presents the real-time selective encryption for baseline and main profile of H.264/AVC, along with an elaborated security analysis. Real-time SE of Jizhun profile of AVS video coding standard of China is presented in Section 7.3. It is followed by concluding remarks in Section 7.4.

7.2 Real-time selective encryption for H.264/AVC

This Section presents a novel method for the protection of H.264/AVC coded video. H.264/AVC supports two types of entropy coding modules. CAVLC is supported in H.264/AVC baseline profile and CABAC is supported in H.264/AVC main profile.

As explained in Chapter 2, H.264/AVS supports variable length coding (VLC) based entropy coding module in baseline profile called CAVLC, while in main profile, it also supports arithmetic coding based entropy coding module called CABAC. SE is performed simultaneously along with entropy coding in entropy coding modules as shown in Fig. 7.1. In baseline profile, SE is performed in CAVLC entropy coding stage (SE-CAVLC). While in main profile, it is performed in CABAC entropy coding stage (SE-CABAC).

SE is performed by using the Advanced Encryption Standard (AES) algorithm with the Cipher Feedback (CFB) mode on a subset of *codewords/binstrings*. For CAVLC, SE is performed on equal length *codewords* from a specific VLC table. In case of CABAC, it is done on equal length *binstrings*.

In our scheme, entropy coding module serves the purpose of encryption cipher without affecting the coding efficiency of video codec by keeping exactly the same bitrate, generating completely compliant bitstream and utilizing negligible computational power. Owing to no escalation in bitrate, our encryption algorithm is better suited for real-time multimedia streaming over heterogeneous networks. It is perfect for playback on hand-held devices because of negligible increase in processing power.

In video encryption, encrypted bitstream compliance is a required feature for some direct operations such as displaying, time seeking and browsing. Encrypted bitstream will be compliant and fulfills real-time constraints if the following three **conditions** are fulfilled:

- To keep the bitrate of encrypted bitstream same as the original bitstream, encrypted *codewords/binstrings* must have the same length as the original *codewords/binstrings*.
- The encrypted *codewords/binstrings* must be valid so that they may be decoded by entropy decoder.
- The decoded value of syntax element from encrypted *codewords/binstrings* must stay in the valid range for that syntax element. Any syntax element which is used for prediction of neighboring MBs should not be encrypted. Otherwise the drift in the value of syntax element will keep on increasing and after a few iterations, value of syntax element will fall outside the valid range and bitstream will be no more decodable.

In each MB, header information is encoded first, which is followed by the encoding of MB data. To keep the bitstream compliant, we cannot encrypt MB header, since it is used for prediction of future MBs. MB data contains NZs and can be encrypted. A MB is further divided into 16 blocks of 4x4 pixels to be processed by IT module. The syntax element *coded block pattern* is used to indicate which 8x8 blocks within a MB contain NZs. The *macroblock mode* (MBmode) is used to indicate whether a MB is *skipped* or not. If MB is not *skipped*, then MBmode indicates the prediction method for a specific MB. For a 4x4 block inside MB, if *coded block pattern* and MBmode are set, it indicates that this block is encoded. Inside 4x4 block, *coded block flag* is the syntax element used to indicate whether it contains NZs (non-zero coefficients) or not. It is encoded first. If it is zero, no further data is transmitted; otherwise, it is followed by encoding of *significant map* in case of CABAC. Finally, the absolute value of each NZ and its sign are encoded. Similar to MB header, header of 4x4 block which includes *coded block flag* and *significant map*, should not be encrypted for the sake of bitstream compliance.

Codewords/binstrings which fulfill the above mentioned conditions constitute the encryption space (ES) for SE-CAVLC and SE-CABAC. Available encryption space is being discussed in Section 7.2.1 and 7.2.2 for SE-CAVLC and SE-CABAC respectively. It is followed by encryption and decryption process of the protected bitstream in Section 7.2.3 and 7.2.4 respectively.

7.2.1 Encryption space (ES) for SE-CAVLC

In CAVLC, five syntax elements are used to code levels and runs as shown in Fig. 7.2. NZs are coded by three syntax elements namely *coeff_ token*, *signs of trailing ones* and *remaining non-zero levels*. Zeros are coded by two syntax elements namely *total*

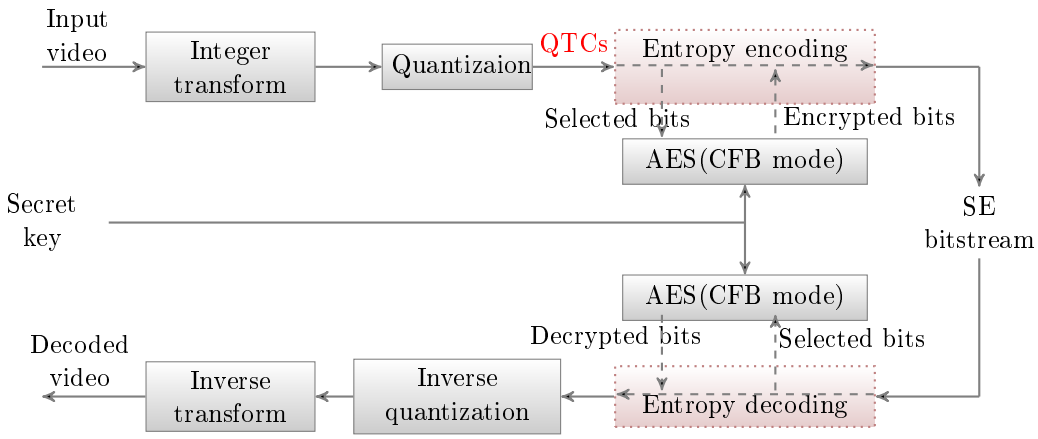


Figure 7.1: Block diagram of encryption and decryption process in H.264/AVC.

no. of zeros and *runs of zeros*. A single syntax element *coeff_token* is used to code total NZs and number of trailing ones (T1's). It is followed by coding of *signs of trailing ones*. *Remaining non-zero levels* are then coded using seven VLC look-up tables either by regular mode or by escape mode as explained in Section 2.3.3. They are mapped to VLC code from a specific VLC look-up table.

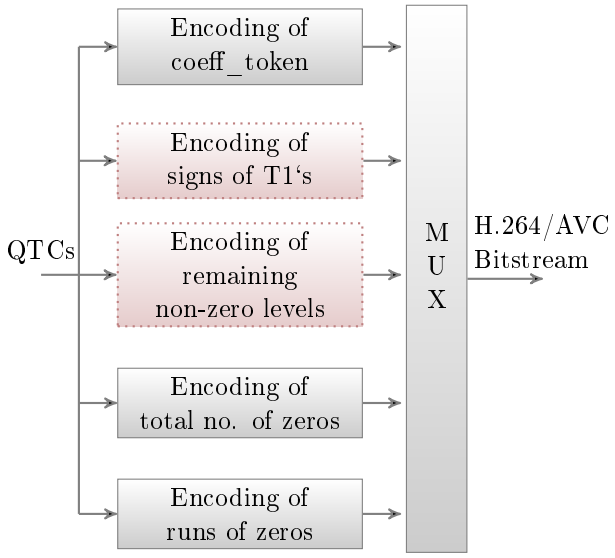


Figure 7.2: Block diagram of CAVLC entropy coding of H.264/AVC. Dotted syntax elements are used for SE-CAVLC.

To keep the bitstream compliant, we cannot encrypt *coeff_token*, *total number of zeros* and *runs of zeros*. Two syntax elements fulfill the above mentioned conditions for encryption. They are *signs of trailing ones* and *sign and magnitude of remaining NZs*, both in regular and escape mode. For the sake of same bitrate, ES of SE-

CAVLC consists of only those NZs having same VLC *codeword* length. CAVLC uses multiple VLC tables with some threshold for incrementing the table as given in equation (7.1). Since the threshold for a specific table is highest possible value possible with that *codeword* length (this is the case when all the suffix bits of the *codeword* are 1), magnitude of encrypted NZ is such that VLC table transition is not affected. VLC codes, having same code length, constitute the ES. For VLCn table, ES is 2^n as given in equation (7.2). For table VLC0, every NZ has different *codeword* length, consequently we cannot encrypt the NZs in table VLC0:

$$TH[0\ldots6] = (0,\ 2,\ 3,\ 6,\ 12,\ 24,\ 48,\ \infty). \tag{7.1}$$

$$ES[0\ldots6] = (1,\ 2,\ 4,\ 8,\ 16,\ 32,\ 64,\ \infty). \tag{7.2}$$

7.2.2 Encryption space (ES) for SE-CABAC

The main difference between SE-CAVLC and SE-CABAC is that in SE-CABAC, SE is not performed on CABAC bitstream. Rather it is performed on *binstrings* which are input to binary arithmetic coder (BAC) as shown in Fig. 7.3. Among all the four *binarization* techniques, the *unary* and *truncated unary* codes have different code lengths for each input value as explained in Section 2.3.4. They do not fulfill the first condition (*i.e.*, length of original and encrypted *binstrings* must be same) and their encryption will change the bitrate of bitstream. Suffix of EGk and the *fixed length* code can be encrypted while keeping the bitrate unchanged. EGk is used for binarization of absolute value of levels and MVDs (motion vector differences) as explained in Section 2.3.4. Number of MVD *binstrings* have the same length and hence, first and second conditions are fulfilled. But owing to the fact that MVDs are part of MB header and are used for prediction of future motion vectors, their encryption does not fulfill third condition and their encryption makes the bitstream non-compliant.

To conclude, the syntax elements which fulfill the criteria for encryption of H.264/AVC compliant bitstream are suffix of EG0 and sign bits of levels. Hence, for each NZ with $|NZ| > 14$, encryption is performed on $l(x)$ of EG0. It is followed by encryption of syntax element *coeff_ sign_ flag* which represents sign of levels of all non-zero levels. The *fixed length* code is used for binarization of syntax elements which belong to MB header and cannot be encrypted.

To keep the bitrate intact, ES for SE-CABAC consists of NZs whose EG0 *binstrings* have same length and sign bit of all NZs as shown in Fig. 7.4. The ES is $2^{log_2(n+1)}$ where n is the maximum possible value by suffix bits of EG0 *i.e.* when all the bits in suffix are 1.

7.2.3 Real-time SE of H.264/AVC

Let us consider $Y_i = X_i \oplus E_k(Y_{i-1})$ as the notation for the encryption of a n bit block X_i, using the secret key k with the AES cipher in CFB mode as given by

Figure 7.3: SE of non-zero coefficients (NZs) in SE-CABAC.

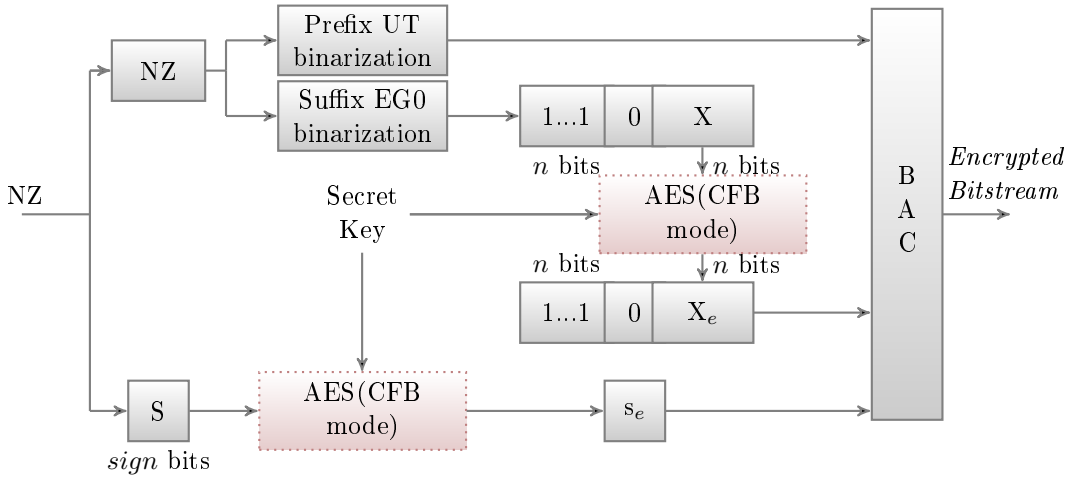


Figure 7.4: Encryption process for NZs and their signs in CABAC of H.264/AVC.

equation 4.2, and performed as described in the scheme from Fig. 4.4. We have chosen to use this mode in order to keep the original compression rate. Indeed, with the CFB mode for each block, the size of the encrypted data Y_i can be exactly the same one as the size of the plaintext X_i. In this mode, the code from the previously encrypted block is used to encrypt the current one as shown in Chapter 4.

The proposed algorithm can be divided into three major steps. First step is the construction of the plaintext X_i and is described in Section 7.2.3.1. It is followed by encryption of X_i to create Y_i which is provided in Section 7.2.3.2. The final step is of substitution of the original *codeword/binstring* by the encrypted *codeword/binstring* is explained in Section 7.2.3.3.

7.2.3.1 The construction of plaintext

As slices are independent coding units, SE should be performed on them independently. In case of SE-CAVLC, the plaintext is created by copying the encryptable bits from CAVLC bitstream to the vector X_i until either X_i is completely filled or slice-boundary comes as shown in Fig. 7.5. In case of SE-CABAC, we perform SE before BAC as shown in Fig. 7.6. In that case, we transform the non-binary syntax elements to *binstrings* through process of binarization and at the same time we fill the X_i with encrypted bits until either the vector X_i is completely filled or the slice

boundary comes. The binarization of many syntax elements at the same time also makes the CABAC coding faster and increases its throughput [Ziauddin 2007].

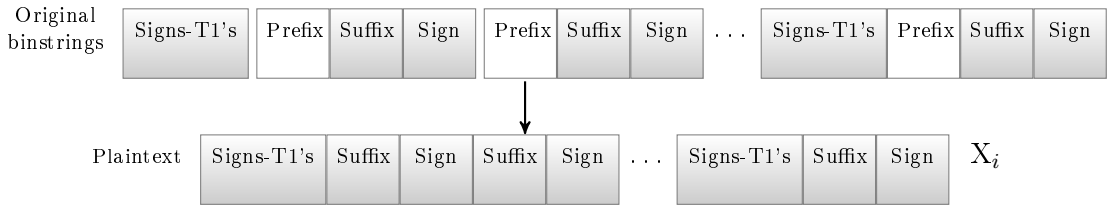


Figure 7.5: Preparation of plaintext for CAVLC.

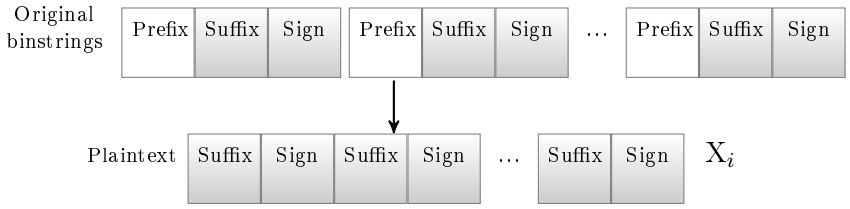


Figure 7.6: Preparation of plaintext for CABAC.

Let C, the length of the vector X_i, is 128 and $L(X_i)$ be the length up to which vector X_i is filled. In case of slice boundary, if $L(X_i) < C$, we apply a padding function $p(j) = 0$, where $j \in \{L(X_i) + 1, \ldots, C\}$, to fill in the vector X_i with zeros up to C bits. Historically, padding was used to increase the security of the encryption, but in here it is used for rather technical reasons [Schneier 1995].

7.2.3.2 Encryption of the plaintext with AES in the CFB mode

In the encryption step with AES in the CFB mode, the previous encrypted block Y_{i-1} is used as the input of the AES algorithm in order to create Z_i, as explained in Section 4.3.2. Then, the current plaintext X_i is XORed with Z_i in order to generate the encrypted text Y_i as given by equation (4.2).

For the initialization, the initialization vector (IV) is created from the secret key k according to the following strategy. The secret key k is used as the seed of the pseudo-random number generator (PRNG). Firstly, the secret key k is divided into 8 bits (byte) sequences. The PRNG produces a random number for each byte component of the key that defines the order of IV formation. Then, we substitute

Y_0 with the IV, and Y_0 is used in AES to produce Z_1.

As illustrated in Fig. 7.7, with the CFB mode of the AES algorithm, the generation of the keystream Z_i depends on the previous encrypted block Y_{i-1}. Consequently, if two plaintexts are identical $X_i = X_j$ in the CFB mode, then always the two corresponding encrypted blocks are different, $Y_i \neq Y_j$.

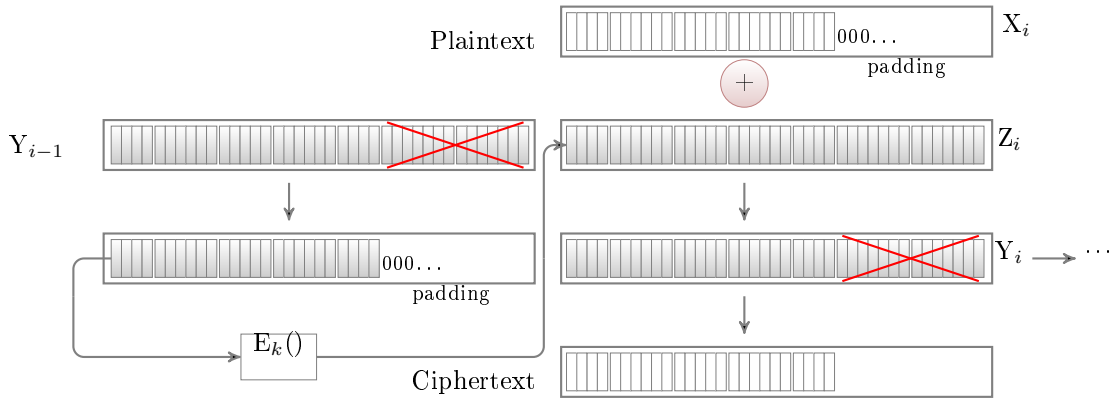


Figure 7.7: Global overview of the proposed SE method. AES cipher has been used in CFB mode for real-time SE.

7.2.3.3 Substitution of the original bitstream

In the process of real-time SE, the third and final step is the substitution of the original X_i by the encrypted Y_i. For SE-CAVLC, CAVLC bitstream is accessed in sequential order as in the first step (construction of the plaintext X_i). Given the length in bits of each encryptable segment $(S_n, S_{n-1}, \ldots, S_1)$, we start substituting the original bits in the bitstream by the corresponding parts of Y_i as shown in Fig. 7.7. For SE-CABAC, *binstrings* are accessed in sequential order and we start substituting the original bits in them by the corresponding parts of Y_i as shown in Fig. 7.7. In case of slice boundaries, the total quantity of replaced bits is $L(X_i)$ and consequently we do not necessarily use all the bits of Y_i.

7.2.4 Decryption process

The decryption process in the CFB mode works as follows. The previous block Y_{i-1} is used as the input to the AES algorithm in order to generate Z_i. By knowing the secret key k, we apply the same function $E_k(\cdot)$ as that used in the encryption stage. The difference is that the input of this process is now the ciphered vector. In case of SE-CAVLC, the ciphered vector is accessed in the sequential way in order to construct the plaintext Y_{i-1} which is then used in the AES to generate the keystream Z_i. The keystream Z_i is then XORed with the current block Y_i to generate X_i, as shown in Fig. 4.4.b. For SE-CAVLC, the resulting plaintext vector is split into

separate segments in order to substitute the signs of trailing ones and suffixes in the ciphered bitstream and to generate the original CAVLC bitstream. Afterward, we apply the entropy decoding and retrieve the quantized DCT coefficients. After the inverse quantization and the inverse DCT we get the decrypted and decoded video frame.

In case of SE-CABAC, the difference is that binary arithmetic decoder is used to transform the SE-CABAC bitstream to encrypted *binstrings* which are then accessed to make the plaintext Y_{i-1}. The plaintext is decrypted and substituted back to generate original *binstrings*. They are then passed through inverse binarization, inverse quantization and inverse DCT steps to get the decrypted and decoded video frame.

7.2.5 Experimental results

In this section, we have used nine benchmark video sequences of Appendix A in QCIF and SD resolutions for analysis of SE-CAVLC and SE-CABAC. We have compressed **100** video frames. For *intra* & *inter* sequence,*intra period* is set to **10**.

In Section 7.2.5.1 we present an analysis of joint SE and H.264/AVC compression which contains analysis of available encryption space and required processing power for SE-CAVLC and SE-CABAC. In Section 7.2.5.2 and Section 7.2.5.3, we compare PSNR and quality when applying SE on *intra* sequences and on *intra* & *inter* sequences respectively.

7.2.5.1 Analysis of joint SE and H.264/AVC compression

We have applied simultaneously our SE and H.264/AVC compression as described in Section 7.2.3, on all the benchmark video sequences. SE-CAVLC and SE-CABAC impart some characteristics to the bitstream. In spatial domain, SE-CAVLC and SE-CABAC videos contain flat regions and change in pixel values mostly occur on MB boundaries. In temporal domain, *luma* and *chroma* values rise up to maximum limit and then come back to minimum values. This cycle keeps on repeating. Owing to this phenomenon, the pixel values change drastically in temporal domain. Lot of transitions are observed in values of color and brightness. This phenomenon can be observed for SE-CAVLC and SE-CABAC in Fig. 7.8 and Fig. 7.9 respectively for QP value 18 for *foreman* video sequence.

In the first set of experiments, we have analyzed the available encryption space (ES) in H.264/AVC bitstreams for both of SE-CAVLC and SE-CABAC. ES is defined as percentage of total bitstream size. MBs that contain many details and texture will have lot of NZs and consequently, will be strongly encrypted. On the other hand, the homogeneous MBs, *i.e.* blocks that contain series of identical pixels, are less ciphered because they contain a lot of null coefficients which are represented by runs in CAVLC and by significant map in CABAC. In Table 7.1, we provide ES for SE-CAVLC and SE-CABAC for different benchmark video sequences for QP value 18. While in Table 7.2, Encryption spaces for various QP values are shown for

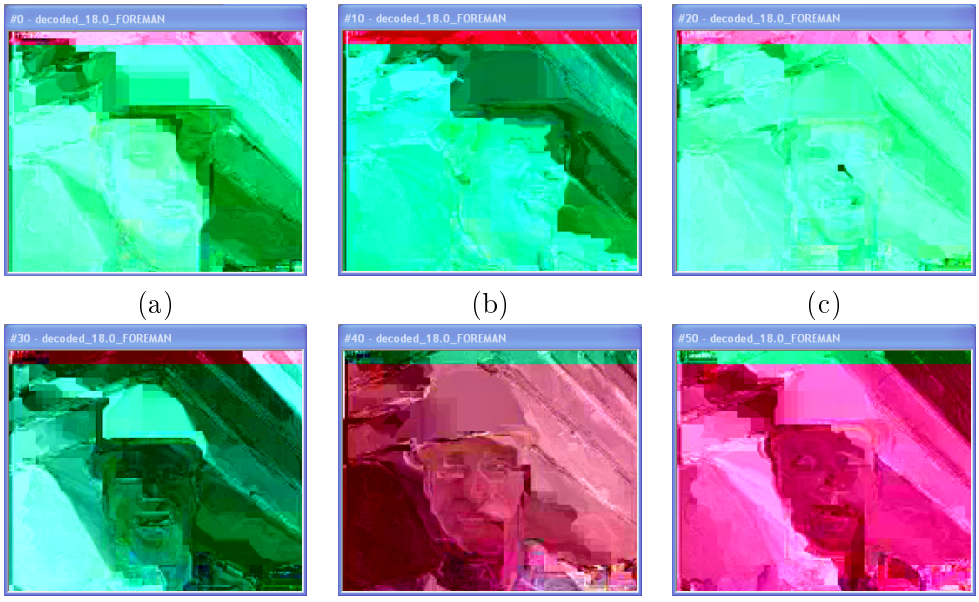


Figure 7.8: Six frames of *foreman* video sequence for SE-CAVLC for QP value 18 for frame: (a) #0, (b) #10, (c) #20, (d) #30, (e) #40, (f) #50.

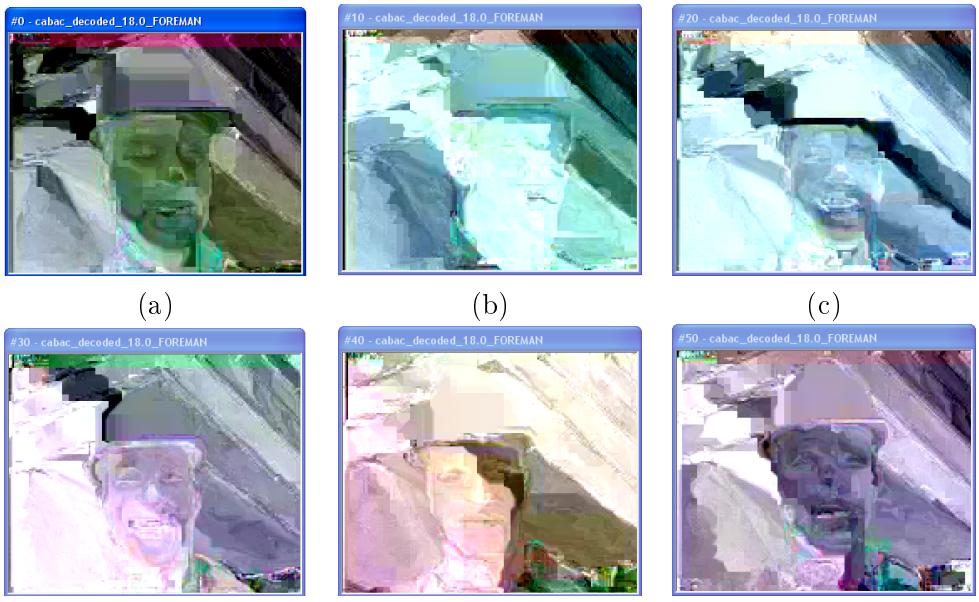


Figure 7.9: Six frames of *foreman* video sequence for SE-CABAC for QP value 18 for frame: (a) #0, (b) #10, (c) #20, (d) #30, (e) #40, (f) #50.

foreman video sequence. Here the average number of encryptable bits per MB are also provided. One can note that ES is inversely proportional to QP value. When

QP value is higher and implicitly the video compression is higher, we are able to encrypt fewer bits in the compressed frame. This is due to the fact that H.264/AVC has lesser number of NZs at higher QP values. From both these tables, it is evident that more ES is available for SE-CAVLC as compared to SE-CABAC. But ES is more affected by change in QP values for SE-CAVLC as compared to SE-CABAC. For example, for *foreman* video sequence, ES varies from 28.55% to 6.70% for SE-CAVLC when QP varies from 12 to 42. For the same QP range, the change in ES for SE-CABAC is from 19.97% to 9.46% as shown in Table 7.2. It is evident from Table 7.1 and 7.2 that PSNR of original H.264/AVC compressed videos are very similar for both CAVLC and CABAC. Hence we will list PSNR of CAVLC videos only in the rest of this section for the sake of comparison.

Table 7.1: Analysis of ES for SE for different benchmark video sequences at QP value 18.

Seq.	SE-CAVLC				SE-CABAC			
	PSNR (dB)	Total Size (Bytes)	ES (%)	Avg. ES Bits/MB	PSNR (dB)	Total Size (Bytes)	ES (%)	Avg. ES Bits/MB
Bus	44.25	1254523	31.05	39	44.24	1255497	19.93	25
City	44.29	1022852	26.41	27	44.27	1024053	19.79	20
Crew	44.82	779480	20.66	16	44.81	777037	18.97	15
Football	44.61	997640	25.33	26	44.59	987936	19.45	19
Foreman	44.38	813195	22.76	19	44.36	806063	18.72	15
Harbour	44.10	1279309	30.49	39	44.09	1268153	20.01	26
Ice	46.47	472573	24.64	12	46.46	469323	17.72	8
Mobile	44.44	1768771	36.17	65	44.43	1753381	19.80	35
Soccer	44.27	922527	23.42	22	44.21	902847	19.94	18

Table 7.2: Analysis of ES for SE over whole range of QP values for *foreman* video sequence.

QP	SE-CAVLC				SE-CABAC			
	PSNR (dB)	Total Size (Bytes)	*ES* (%)	Avg. *ES* Bits/MB	PSNR (dB)	Total Size (Bytes)	*ES* (%)	Avg. *ES* Bits/MB
12	50.07	1260001	28.55	36	50.05	1257024	19.97	25
18	44.38	813195	22.76	19	44.36	806063	18.72	15
24	39.43	478496	17.13	8	39.42	464794	17.61	8
30	35.08	268012	13.24	4	35.08	255287	15.65	4
36	31.04	145736	9.88	1	31.06	134143	12.22	2
42	27.23	88333	6.70	1	27.35	70616	9.46	1

Processing power required for a specific algorithm is very critical for any real-time algorithm. Our proposed real-time SE schemes take minimal processing power. Table 7.3 gives a detailed overview of the required processing power for *intra* and

intra & *inter* video sequences at QP value 18. Increase in computation time for encoder is less than 0.4% for both of SE-CAVLC and SE-CABAC while it is below than 3% on decoder side for *intra* & *inter* sequence.

Fig. 7.10.a and 7.10.b show the frame-wise analysis of increase in processing power for SE-CABAC at QP value 18 for *foreman*. For experimentation, 2.1 GHz Intel Core 2 Duo T8100 machine with 3072 MB RAM has been used. For *intra* & *inter* sequence encoding of 100 frames with *intra period* 10, it took 4372.5 seconds and 4381.3 seconds for CABAC and SE-CABAC respectively. While it took 2.005 seconds and 2.045 seconds for CABAC and SE-CABAC decoding. It is a negligible increase in processing power and can be managed well even by hand-held devices. It is important to note that required processing power for SE-CABAC is less than SE-CAVLC owing to two reasons. First, ES of SE-CABAC is lesser than that of SE-CAVLC as shown in Table 7.1 and Table 7.2. Second, CABAC takes a lot more processing power than CAVLC. So increase in processing power because of encryption will be lower in terms of percentage. Thus, SE-CAVLC and SE-CABAC is possible in real-time along with compression.

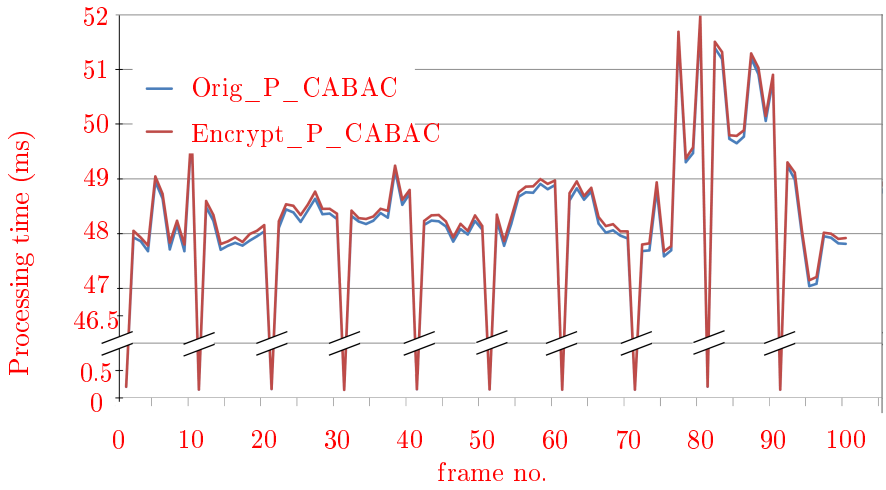
Table 7.3: Analysis of increase in processing power for SE-CAVLC and SE-CABAC at QP value 18.

Seq.	SE-CAVLC				SE-CABAC			
	encoder		decoder		encoder		decoder	
	I (%)	I+P (%)	I (%)	I+P (%)	I (%)	I+P (%)	I (%)	I+P (%)
Bus	0.69	0.31	3.77	2.7	0.57	0.25	3.37	2.3
City	0.50	0.26	3.36	2.4	0.44	0.23	3.06	2.1
Crew	0.31	0.15	2.52	1.5	0.29	0.14	2.22	1.2
Football	0.41	0.23	3.46	2.4	0.31	0.18	3.26	2.2
Foreman	0.47	0.23	3.19	2.2	0.41	0.20	2.99	2.0
Harbour	0.55	0.30	3.65	2.7	0.47	0.26	3.25	2.3
Ice	0.41	0.21	3.16	2.1	0.33	0.17	2.96	1.9
Mobile	0.76	0.35	4.33	3.3	0.72	0.33	4.03	3.0
Soccer	0.44	0.21	3.17	2.2	0.38	0.18	2.87	1.9

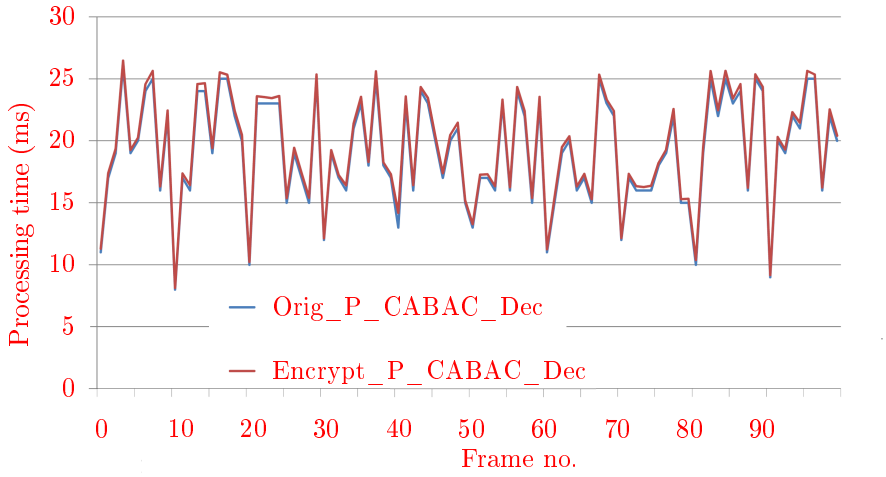
7.2.5.2 PSNR and quality of SE-CAVLC & SE-CABAC for *intra* frames

PSNR is a widely used objective video quality metric. However, it does not perfectly correlate with a perceived visual quality due to non-linear behavior of human visual system. Structural similarity index (SSIM) [Wang 2004a] takes into account the structural distortion measurement, since human vision system is highly specialized in extracting structural information from the viewing field. In this section, we have used PSNR and SSIM for measurement of visual protection of encrypted *intra* sequences.

For the visual analysis, Fig. 7.11 and Fig. 7.12 show the encrypted frame #0



(a)



(b)

Figure 7.10: Framewise time taken by SE-CABAC of *foreman* video sequence for *intra* & *inter* sequence at QP value 18 with *intra period* 10 during: (a) Encoding, (b) Decoding.

of *foreman* at different QP values for SE-CAVLC and SE-CABAC respectively. In H.264/AVC, blocks on the top array are predicted only from left while blocks on left are always predicted from top. Owing to this prediction, a band having width of 8 pixels at top of video frames can be observed for both of SE-CAVLC and SE-CABAC while this band has width of 4 pixels on left of video frames as shown in Fig. 7.11 and Fig. 7.12. The average PSNR values of *foreman* is given in Table 7.4 over whole QP range. It is also compared with the PSNR obtained for the same video sequence without encryption. One can note that whatever is the QP value, the quality of the encrypted video remains in the same lower range.

Table 7.5 compares the average PSNR of 100 *intra* frames of all benchmark video sequences at QP value 18 without encryption and with SE. Average PSNR value of *luma* for all the sequences at QP value 18 is 9.49 *dB* for SE-CAVLC and 9.80 *dB* for SE-CABAC. It confirms that this algorithm works well for various combinations of motion, texture and objects for *intra* frames. It is also evident in frame-wise PSNR of *luma* of *intra* frames of *foreman* video sequence as shown in Fig. 7.13.

With the increase in processing power and memory size, video resolutions are available up to High definitions now. To show the validity of our scheme at higher resolutions, Table 7.6 contains the experimental results of SE of *intra* sequence for SD resolution. Here, Average PSNR value of *luma* is 9.82 *dB* for SE-CAVLC and 9.83 *dB* for SE-CABAC, which is almost the same as that of QCIF resolution. It is evident that this algorithm is capable to encrypt important visual information at all resolutions. For the rest of the section, we present analysis for QCIF resolution only, since more benchmark video sequences are available to us in this resolution.

Table 7.4: PSNR comparison for *intra* frames without encryption and with SE for *foreman* at different QP values.

QP	PSNR (Y) (dB)			PSNR (U) (dB)			PSNR (V) (dB)		
	Orig.	SE CAVLC	SE CABAC	Orig.	SE CAVLC	SE CABAC	Orig.	SE CAVLC	SE CABAC
12	50.07	8.61	8.43	50.00	19.78	24.09	50.79	9.57	22.58
18	44.38	8.67	8.58	45.68	24.14	24.40	47.57	10.16	22.10
24	39.43	8.71	8.72	41.93	26.39	24.35	44.19	24.91	22.84
30	35.08	9.43	8.69	39.75	27.45	24.58	41.41	25.36	23.64
36	31.04	9.37	8.53	37.74	28.12	24.93	38.62	24.78	23.16
42	27.23	9.45	8.67	36.23	25.51	24.91	36.86	24.59	24.02

SSIM is another quality metric which is very close to HVS. It takes into account the structural information, as explained in Section 2.6. Table 7.7 shows the SSIM values of *luma* of benchmark video sequences without encryption and with SE. Results verify that the proposed scheme has distorted the structural information present in the original video. Average SSIM value of video sequences without encryption is 0.993, while it is 0.164 and 0.180 for SE-CAVLC and SE-CABAC respectively. Fig. 7.14 shows the frame-wise SSIM of *luma* of *foreman* video sequence

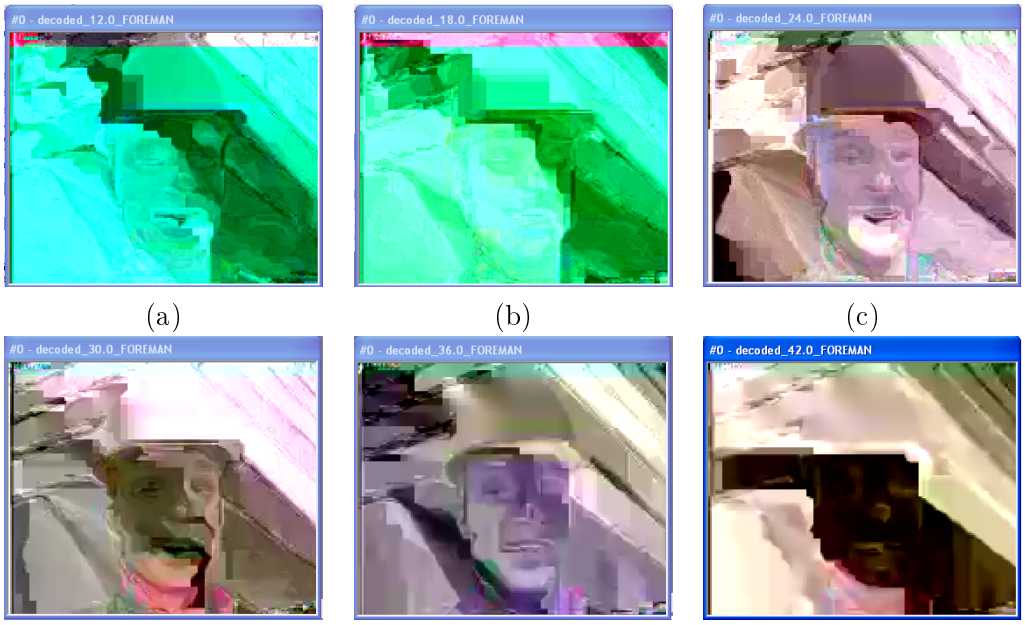


Figure 7.11: Decoding of SE-CAVLC frame #0 of *foreman* video sequence with QP value: (a) 12, (b) 18, (c) 24, (d) 30 (e) 36, (f) 42.

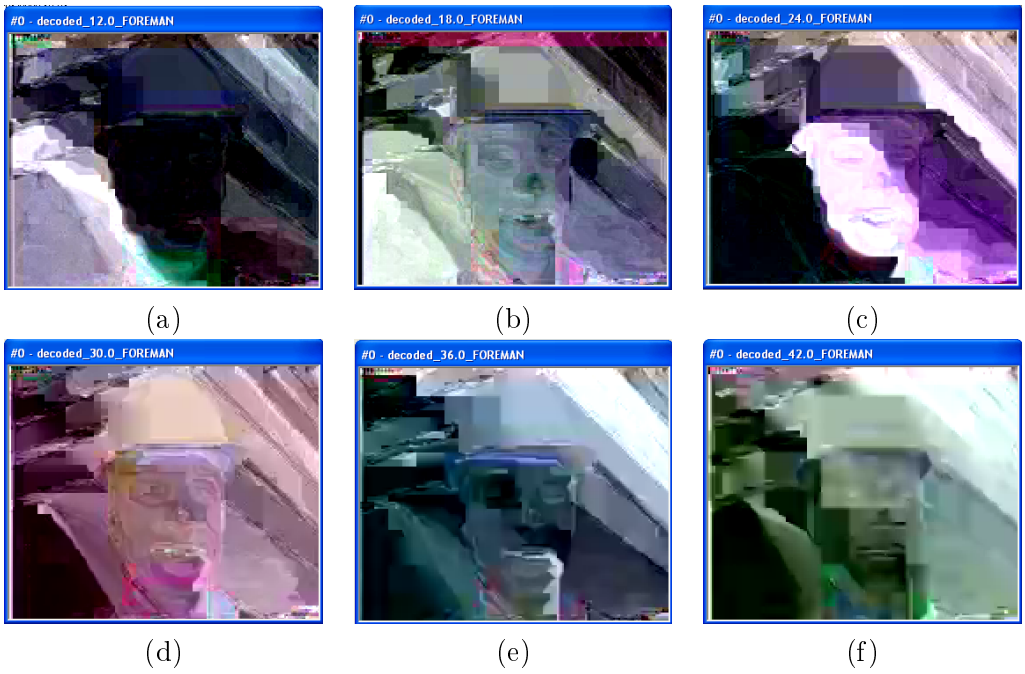


Figure 7.12: Decoding of SE-CABAC frame #0 of *foreman* video sequence with QP value: (a) 12, (b) 18, (c) 24, (d) 30, (e) 36, (f) 42.

for *intra* frames. It is important to note that SSIM value of complex video sequences is less than that of simple video sequences.

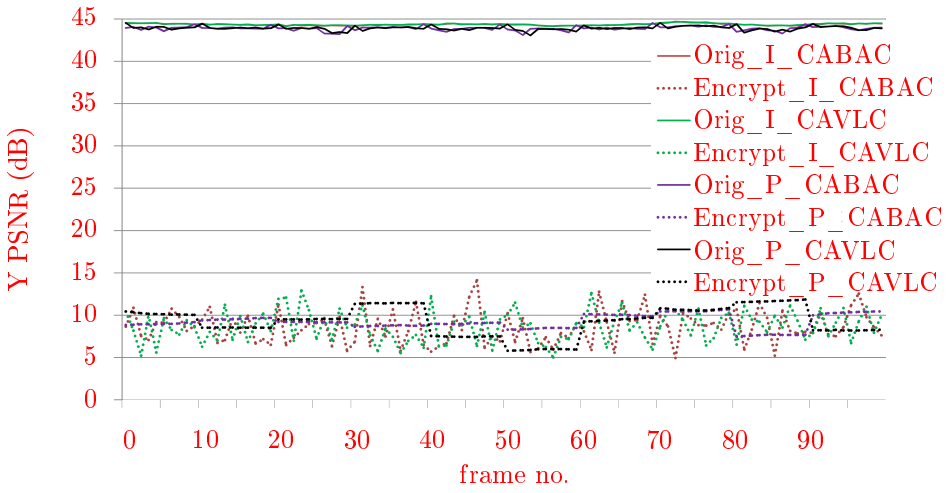


Figure 7.13: Frame wise PSNR of *intra* and *intra* & *inter* frames for *foreman* for SE-CAVLC and SE-CABAC at QP value 18.

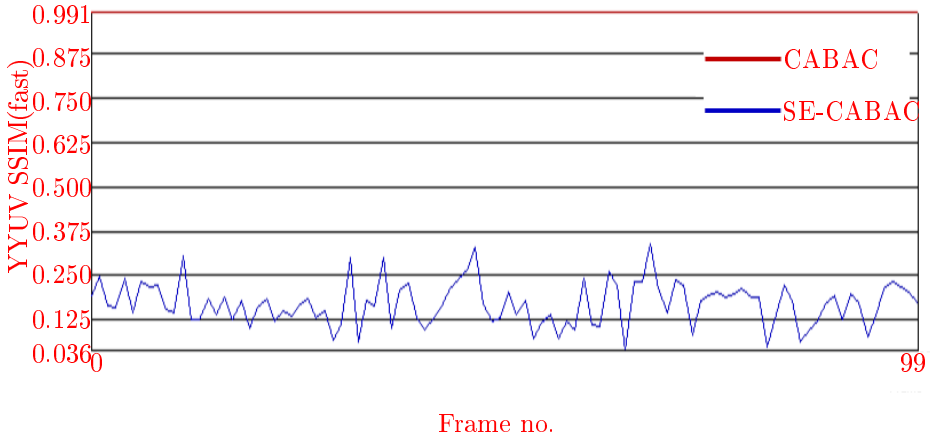


Figure 7.14: Frame wise SSIM of *intra* frames for *foreman* for SE-CABAC at QP value 18.

Table 7.5: PSNR comparison for *intra* frames without encryption and with SE at QP value 18.

Seq.	PSNR (Y) (dB)			PSNR (U) (dB)			PSNR (V) (dB)		
	Orig.	SE CAVLC	SE CABAC	Orig.	SE CAVLC	SE CABAC	Orig.	SE CAVLC	SE CABAC
Bus	44.25	7.90	8.18	45.22	26.82	24.95	46.55	26.65	27.25
City	44.29	10.90	11.23	45.84	31.89	30.27	46.83	33.47	31.80
Crew	44.82	8.96	9.90	45.84	23.99	23.45	45.70	19.74	19.79
Football	44.61	11.48	11.49	45.77	14.85	14.39	46.05	24.28	23.59
Foreman	44.38	8.67	8.58	45.68	24.14	24.40	47.57	10.16	22.10
Harbour	44.10	9.25	9.50	45.61	27.07	24.61	46.67	33.25	31.31
Ice	46.47	10.59	10.40	48.81	24.26	25.58	49.28	16.86	20.39
Mobile	44.44	8.32	8.29	44.15	10.44	13.08	44.06	9.58	10.97
Soccer	44.27	9.34	10.61	46.62	22.10	19.73	47.93	28.21	24.41
Avg.	44.63	9.49	9.80	45.95	22.84	22.27	46.74	22.47	23.51

Table 7.6: PSNR comparison for *intra* frames without encryption and with SE at QP value 18 for **SD** resolution).

Seq.	PSNR (Y) (dB)			PSNR (U) (dB)			PSNR (V) (dB)		
	Orig.	SE CAVLC	SE CABAC	Orig.	SE CAVLC	SE CABAC	Orig.	SE CAVLC	SE CABAC
City	44.65	9.94	10.12	47.82	27.34	26.20	49.07	31.37	29.92
Crew	45.15	9.16	9.08	46.56	24.52	22.80	47.74	20.14	19.97
Harbour	44.54	9.35	9.37	47.48	22.91	22.92	48.73	28.79	26.81
Ice	46.17	10.67	10.38	51.50	27.79	27.72	52.01	25.04	26.09
Soccer	45.12	9.96	10.19	47.68	18.36	18.02	49.21	26.68	24.08
Avg.	45.13	9.82	9.83	48.21	24.18	23.53	49.35	26.40	25.37

Table 7.7: **SSIM** comparison of *luma* of *intra* frames without encryption and with SE at QP value 18.

Seq.	ORIG-CAVLC	SE-CAVLC	ORIG-CABAC	SE-CABAC
Bus	0.995	0.069	0.994	0.064
City	0.994	0.115	0.994	0.093
Crew	0.991	0.184	0.991	0.153
Football	0.991	0.219	0.991	0.184
Foreman	0.990	0.198	0.990	0.165
Harbour	0.998	0.047	0.998	0.038
Ice	0.990	0.419	0.990	0.398
Mobile	0.998	0.040	0.998	0.356
Soccer	0.988	0.185	0.988	0.171
Avg.	0.993	0.164	0.993	0.180

7.2.5.3 PSNR of SE-CAVLC & SE-CABAC for *intra* & *inter* frames

Table 7.8: PSNR comparison for *intra* & *inter* frames without encryption and with SE for *foreman* at different QP values.

QP	PSNR (Y) (dB)			PSNR (U) (dB)			PSNR (V) (dB)		
	Orig.	SE CAVLC	SE CABAC	Orig.	SE CAVLC	SE CABAC	Orig.	SE CAVLC	SE CABAC
12	49.55	8.73	8.11	49.89	18.35	22.98	50.63	10.41	21.63
18	43.93	9.14	10.44	45.53	23.56	23.87	47.56	8.03	23.19
24	38.92	9.60	9.72	42.04	26.93	24.87	44.27	25.77	24.98
30	34.60	9.24	9.25	39.84	28.61	24.95	41.54	26.63	24.03
36	30.72	10.09	8.19	37.91	28.45	24.28	38.75	22.78	23.36
42	26.95	9.44	8.64	36.30	26.46	26.82	36.92	25.60	24.65

Table 7.9: Comparison of PSNR without encryption and with SE for *intra* & *inter* frames at QP value 18.

Seq.	PSNR (Y) (dB)			PSNR (U) (dB)			PSNR (V) (dB)		
	Orig.	SE-CAVLC	SE-CABAC	Orig.	SE-CAVLC	SE-CABAC	Orig.	SE-CAVLC	SE-CABAC
Bus	43.73	7.58	7.72	45.10	27.15	25.42	46.42	24.73	27.01
City	43.81	11.42	11.14	45.73	32.47	30.16	46.76	32.53	31.66
Crew	44.46	8.97	10.00	45.81	25.09	21.98	45.73	19.63	20.18
Football	44.16	12.13	11.28	45.72	14.31	14.58	46.06	24.77	24.27
Foreman	43.93	9.14	10.44	45.53	23.56	23.87	47.56	8.03	23.19
Harbour	43.71	9.46	9.78	45.45	24.53	22.93	46.58	33.87	31.67
Ice	46.14	10.93	10.38	48.61	23.63	25.29	49.14	19.17	19.71
Mobile	43.85	8.44	8.84	44.16	10.09	12.48	44.07	9.61	11.85
Soccer	43.56	9.65	10.56	46.47	21.83	20.76	47.76	27.40	22.24
Avg.	44.15	9.75	10.02	45.84	22.52	21.94	46.68	22.19	23.53

It is important to confirm the validity of the proposed schemed for *intra* & *inter* sequence, since video data normally consists of an *intra* frame and a trail of *inter* frames. *Intra* frames are inserted periodically to restrict the drift because of lossy compression and rounding errors. In this section, visual protection of encrypted video sequences of *intra* & *inter* frames is only presented by PSNR. SSIM quality metric has very low values even for *intra* sequence and is not given here for the sake of brevity. Results shown in Table 7.8 verify the effectiveness of our scheme over the whole range of QP values for *foreman* video sequence. Table 7.9 verifies the performance of our algorithm for all video sequences for *intra* & *inter* sequence at QP value 18. Average PSNR of *luma* for all the sequences is 9.75 *dB* and 10.02 *dB* for SE-CAVLC and SE-CABAC respectively.

Fig. 7.13 shows the frame-wise PSNR of *luma* of *foreman* video sequence for *intra* & *inter*. Here PSNR of SE-CAVLC and SE-CABAC remains almost the same for sequence of P frames and changes at every *intra* frame, thus producing a staircase graph.

7.2.6 Security analysis

Security analysis is really important for the validity of any SE scheme. Analysis of entropy and local standard deviation is presented in Section 7.2.6.1. It is followed by correlation analysis in Section 7.2.6.2. Key sensitivity test and removal of encrypted data attacks are then presented in Section 7.2.6.3 and Section 7.2.6.4 respectively.

7.2.6.1 Analysis of entropy and local standard deviation

The security of the encrypted image can be measured by considering the variations (local or global) in the protected image. Entropy is a statistical measure of randomness or disorder of a system which is mostly used to characterize the texture in the input images. Considering this, the information content of image can be measured with the entropy $H(X)$ and local standard deviation $\sigma(j)$. If an image has 2^k gray levels α_i with $0 \leq i \leq 2^k$ and the probability of gray level α_i is $P(\alpha_i)$, and without considering the correlation of gray levels, the 1^{st} order entropy $H(X)$ is defined as:

$$H(X) = -\sum_{i=0}^{2^k-1} P(\alpha_i) log_2(P(\alpha_i)). \tag{7.3}$$

If the probability of each gray level in the image is $P(\alpha_i) = \frac{1}{2^k}$, then the encryption of such image is robust against statistical attacks of 1^{st} order, and thus $H(X) = log_2(2^k) = k$ bits/pixel. In the image the information redundancy r is defined as:

$$r = k - H(X). \tag{7.4}$$

Similarly the local standard deviation $\sigma(j)$ for each pixel $p(j)$ taking account of its neighbors to calculate the local mean $\overline{p(j)}$, is given as:

$$\sigma(j) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(p(i) - \overline{p(j)})}, \tag{7.5}$$

where m is the size of the pixel block to calculate the local mean and standard deviation, and $0 \leq j < M$, if M is the image size.

In case of full encryption, entropy $H(X)$ is maximized with high values of local standard deviation. But in case of SE-CAVLC and SE-CABAC, the video frame is transformed to flat regions with blocking artifacts as depicted in Fig. 7.11 and Fig. 7.12. It is generally owing to variations in pixel values at MB boundaries. For all the benchmark sequences, the average information redundancy r for SE-CAVLC and SE-CABAC sequences is 0.94 and 0.55 respectively, while it is 1.11 for all the

original sequences. Despite the fact that SE-CAVLC and SE-CABAC transform the video frames into flat regions, the entropy of the encrypted video sequences from equation (7.3) is higher as compared to original sequences.
These flat regions are because of two reasons. Firstly, flat regions are due to the fact that prediction is performed from edge pixels of neighboring MBs. Secondly, pixels are either with very high value(bright video frame) or very low value (dark video frame) in SE video frame. This is owing to the fact that during reconstruction pixel value are clipped to 255 if they are greater than it and to 0 if they are below this lower range. So if many pixels have value beyond the upper or lower range, all of them will be clipped to the same value, thus creating a flat region which is either dark or bright. Based on this analysis, the statistical characteristics of SE-CAVLC and SE-CABAC bitstreams vary from full encryption systems.

From equation (7.5), we also analyzed the local standard deviation σ for each pixel while taking into account its neighbors. In Table 7.11, the mean local standard deviation for *foreman* sequence at different QP values is given. For all benchmark video sequences, the mean local standard deviation of *luma* equals to 69.15 and 61.48 for the SE-CAVLC and SE-CABAC bitstreams respectively, where the mean local standard deviation is less than 10 gray levels for the original benchmark sequences. One can note that local standard deviation of encrypted sequences is higher than original sequences.

In general, encryption tends to makes the encrypted symbols uniformly distributed. Fig. 7.15 shows the histogram of the original and the encrypted NZs of *foreman* video sequence for SE-CAVLC. Here we can see that SE has given an effect of staircase because of treating the coefficients with equal probability in the same interval.

Table 7.10: Standard deviation for SE of *foreman* video sequence at different QP values.

	STD-CAVLC		STD-CABAC	
QP	Orig.	SE-CAVLC	Orig.	SE-CABAC
12	6.75	71.49	7.02	69.69
18	7.21	73.23	7.53	59.97
24	8.57	91.98	8.63	84.55
30	6.35	35.99	6.71	57.87
36	6.90	47.42	6.93	68.04
42	7.91	75.26	8.11	71.17

7.2.6.2 Correlation of adjacent pixels

Visual data is highly correlated i.e. pixels values are highly probable to repeat in horizontal, vertical and diagonal directions. A correlation of a pixel with its neighboring pixel is then given by a tuple (x_i, y_i) where y_i is the adjacent pixel of

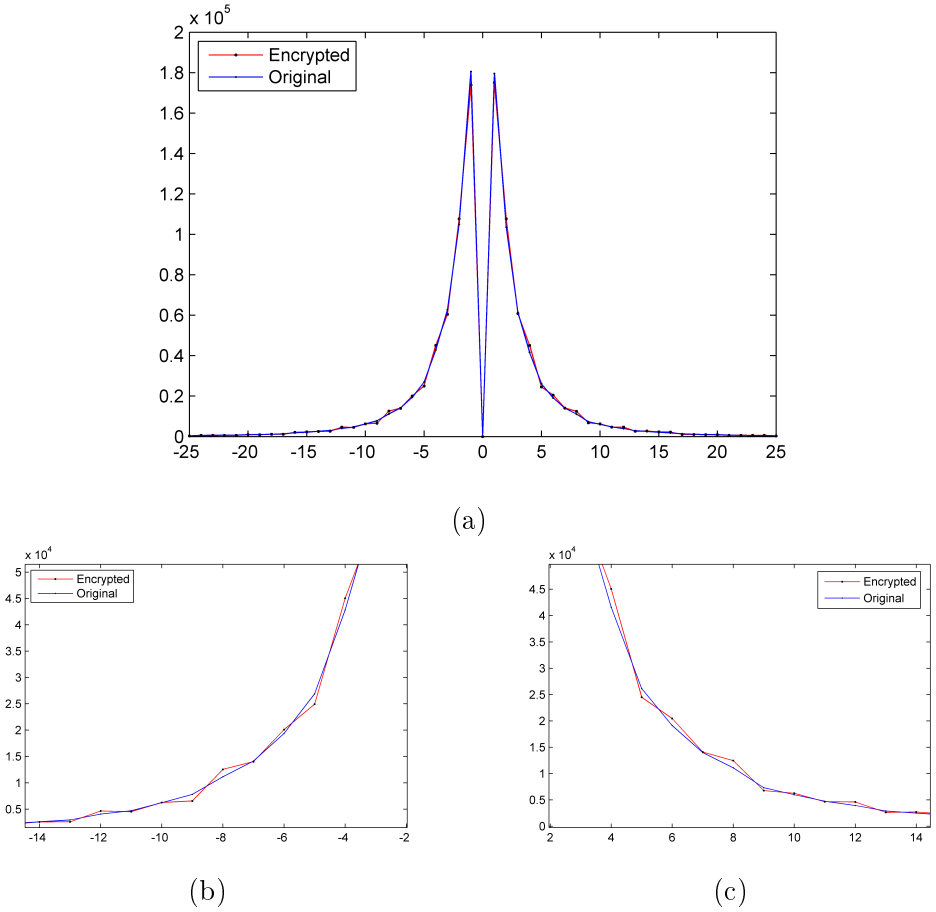


Figure 7.15: Histograms of original and SE-CAVLC NZs: (a) Complete graph, (b) Zoomed graph of negative x-axis, (c) Zoomed graph of positive x-axis.

Table 7.11: Standard deviation for SE of *foreman* video sequence at different QP values.

	STD-CAVLC		STD-CABAC	
QP	Orig.	SE-CAVLC	Orig.	SE-CABAC
12	6.75	71.49	7.02	69.69
18	7.21	73.23	7.53	59.97
24	8.57	91.98	8.63	84.55
30	6.35	35.99	6.71	57.87
36	6.90	47.42	6.93	68.04
42	7.91	75.26	8.11	71.17

x_i. Since there are always three directions in images *i.e.* horizontal, vertical and diagonal, so we can define correlation direction between any two adjacent pixels as:

$$corr_{(x,y)} = \frac{1}{n-1}\sum_{0}^{n}(\frac{x_i - \overline{x_i}}{\sigma_x})(\frac{y_i - \overline{y_i}}{\sigma_y}), \tag{7.6}$$

where n represents the total number of tuples (x_i, y_i), $\overline{x_i}$ and $\overline{y_i}$ represent the local mean and σ_x and σ_y represent the local standard deviations respectively.

Owing to the flat regions in SE-CAVLC and SE-CABAC video sequences, the correlation values in these sequences will be higher as compared to original image which contain texture and edges. For all the benchmark sequences, the average horizontal correlation coefficient is 0.88 and 0.87 for the SE-CAVLC and SE-CABAC respectively, while it is 0.80 for the original sequences.

7.2.6.3 Key sensitivity test

Robustness against cryptanalyst can be improved if the cryptosystem is highly sensitive towards the key. The more the visual data is sensitive towards the key, the more we would have data randomness. For this purpose, a key sensitivity test is assumed where we pick one key and then apply the proposed technique for encryption and then make a one bit change in the key and decode the bitstream. Numerical results show that the proposed technique is highly sensitive towards the key change, that is, a different version of encrypted video sequence is produced when the keys are changed, as shown in Fig. 7.16. PSNR of *luma* of decrypted frames with 1-bit different key is 10.39 *dB* and 8.31 *dB* for SE-CAVLC and SE-CABAC respectively, as shown in Table 7.12. It lies in the same lower range as decoded frames without decryption.

(a) (b) (c)

Figure 7.16: Key sensitivity test for encrypted frame #0 of *foreman* video sequence for QP value 18. Encrypted frames are decrypted and decoded with: (a) original key, (b) 1-bit different key(SE-CAVLC), (c) 1-bit different key(SE-CABAC).

7.2.6.4 Removal of encrypted data attack

In another experiment we have replaced the encrypted bits with constant values in order to measure the strength of SE-CAVLC and SE-CABAC proposed method as described in [Said 2005]. Here we have used frame #0 of *foreman* video sequence

Table 7.12: Key sensitivity test of SE-CAVLC and SE-CABAC encrypted video for frame #0 *foreman* video sequence for QP value 18.

	PSNR (Y) (dB)	PSNR (U) (dB)	PSNR (V) dB)
Original key	44.60	45.73	47.35
SE-CAVLC (1-bit different key)	10.39	24.46	14.02
SE-CABAC (1-bit different key)	8.31	25.13	24.82

(a) (b) (c) (d)

Figure 7.17: Attack in the selectively encrypted image by removing the encrypted data: (a) SE-CAVLC encrypted image {Y, U, V} = {10.01, 26.86, 25.24} dB, (b) SE-CAVLC attacked image {Y, U, V} = {8.87, 27.3, 26.3} dB, (c) SE-CABAC encrypted image {Y, U, V} = {8.20, 17.95, 24.53} dB, (d) SE-CABAC attacked image {Y, U, V} = {7.72, 28.6, 24.6} dB.

with QP value 24. Fig. 7.17 shows both encrypted and attacked video frames for SE-CAVLC and SE-CABAC. For example, Fig. 7.17.a shows SE-CAVLC video frame with PSNR of 10.01 *dB* for *luma*. If we set the encrypted bits of all NZs to zero, we get the video frame illustrated in Fig. 7.17.b with *luma* PSNR of 8.87 *dB*. Similarly, Fig. 7.17.c shows SE-CABAC video frame having PSNR of 8.20 *dB* while the attacked SE-CABAC video frame has PSNR of 7.72 *dB* as shown in Fig. 7.17.d.

7.2.7 Comparative evaluation

For the sake of comparative evaluation of our scheme, we have compared it with six other recent techniques, which includes scrambling [Dufaux 2008], NAL unit encryption [Li 2008], MB header encryption [Lian 2007], reversible ROI encryption [Carrillo 2009], *intra* frame encryption [Abomhara 2010] and multiple Huffman table permutation [Wu 2005a]. These techniques are different from each other in several aspects e.g. working domain (pixel, transform or bitstream) and encryption algorithm (pseudorandom permutation, stream cipher or AES). The comparison has been made based on several important characteristics of SE systems and is summarized in Table 7.13.

Encryption algorithm used in SE scheme is of vital importance for the security level. AES has the highest security among all the known ciphers and our proposed scheme utilizes AES. Among the recent techniques, AES has been used only in [Abomhara 2010] but their SE scheme is very naive and encrypts bitstream of

intra frames only.

Selective encryption should not result in increase of bitrate. For example, if a video for 3G wireless connection has bitrate of 384 kbps. Its encrypted version should have the same bitrate. Otherwise it cannot not be played back on 3G connection. Our scheme keeps the bitrate intact. It is in contrast to other schemes which either allow increase in bitrate [Dufaux 2008, Carrillo 2009, Wu 2005a], or use stream cipher for the sake of same bitrate [Li 2008, Lian 2007], thus compromising on the security of the system.

Format compliance is another important aspect for encrypted video and is required for rate adaptation for multimedia on heterogeneous networks [Li 2001], DC image extraction for multimedia content searching [Yeo 1995]. It is also required for network layer decisions [Wu 2001] and unequal error adaptation [Wang 2000]. Most of the schemes are not format complaint and their encrypted bitstreams cannot be decoded by reference decoder except SE schemes which work in pixel domain [Carrillo 2009] and transform domain [Dufaux 2008].

Our SE-CABAC scheme is the first format compliant technique which is for arithmetic coding based entropy coding module, while keeping the bitrate unchanged. Recent encryption techniques for arithmetic coding [Jiangtao 2006, Grangetto 2006] are not format complaint and require lot of processing power.

To summarize, our proposed schemes (SE-CAVLC and SE-CABAC) meet all the requirements of an integrated crypto-compression systems. Our proposed system is fully compliant to H.264/AVC decoder, with no change in bitrate and has the security of AES cipher.

Table 7.13: Comparison of our proposed schemes with other recent methods.

Video Selective Encryption Scheme	Format compliant	Robust to transcoding	Domain	Bitrate increase	Compression independent	Encryption algorithm
Scrambling for privacy protection [Dufaux 2008]	Yes	No	Transform	Yes	Yes	Pseudorandom sign inversion
NAL unit encryption [Li 2008]	No	No	Bitstream	No	No	Stream Cipher
MB header encryption [Lian 2007]	No	No	Transform	No	No	Stream Cipher
Reversible encryption of ROI [Carrillo 2009]	Yes	Yes	Pixel	Yes	Yes	Pseudorandom permutations
I frame encryption [Abomhara 2010]	No	No	Bitstream	No	No	AES
Multiple Huffman tables [Wu 2005a]	No	No	Bitstream	Yes	No	Huffman Table permutations
Our scheme	Yes	No	Bitstream*	No	No	AES (CFB mode)

* For SE-CAVLC, bitstream is encrypted, while for SE-CABAC, binstrings are encrypted as explained in Section 7.2.2.

7.2.8 Synopsis

In Section 7.2, an efficient SE system has been proposed for H.264/AVC video codec for CAVLC and CABAC. The SE is performed in the entropy coding stage of the

H.264/AVC using the AES encryption algorithm in the CFB mode. In this way the proposed encryption method does not affect the bitrate and the H.264/AVC bitstream compliance. The SE is performed in CAVLC *codewords* and CABAC *binstrings* such that their encrypted version is valid *codewords/binstrings* thereafter having exactly the same length. Experimental analysis has been presented for *intra* and *inter* frames. The proposed scheme can be used for B (bi-directionally predicted) frames without any modification, since B frames are also *inter* frames but have bidirectional prediction.

7.3 Real-time selective encryption of AVS

Audio Video Coding Standard (AVS) [Fan 2004] is the emerging video coding standard of China, as presented in Chapter 2. It has 3% less performance as compared to H.264/AVC, with much less complexity as compared to the latter, as demonstrated on HD progressive-scan sequences by [Wang 2004a]. In AVS part-2 Jizhun profile, it supports VLC based entropy coding module called C2DVLC. In C2DVLC, 2D VLC tables are used which codes (L_i, R_i) pair as a single event. In this Section, real-time SE of AVS is presented along with its real-time implementation. It is performed in the C2DVLC module of video codec. For this purpose, AES cipher with the Cipher Feedback (CFB) mode is used on a subset of codewords. C2DVLC serves the purpose of encryption step without affecting the coding efficiency of AVS by keeping the bitrate unchanged, generating completely compliant bit stream and utilizing negligible computational power.

In contrast to SE-CAVLC of H.264/AVC, certain constraints have to be fulfilled for real-time format compliant SE of C2DVLC (SE-C2DVLC). The main differences between CAVLC of H.264/AVC and C2DVLC of AVS from selective encryption point of view are that C2DVLC codewords are not contiguous and they do not use the full code-space.

SE of C2DVLC codewords is being presented in Section 7.3.1, while the proposed schemes for its real-time implementation are discussed in Section 7.3.2.

7.3.1 Selective encryption of C2DVLC (SE-C2DVLC))

We want SE-C2DVLC scheme which fulfills real-time constraints (same bitrate, format compliance), while utilizing minimal processing power. To keep the bitrate unchanged along with encrypted bitstream format compliance, we perform encryption of C2DVLC while fulfilling the following **constraints**:

- In the (L_i, R_i) pair, only L_i can be encrypted. R_i value must not be changed otherwise the bitstream will not be decodable.
- From equation (2.8), the encrypted symbol should be such that L_{max} remains in the same interval, thus selecting the same context for the next (L_i, R_i).
- For Exp-Golomb coding, the length of the encrypted codeword must be equal to that of original codeword.

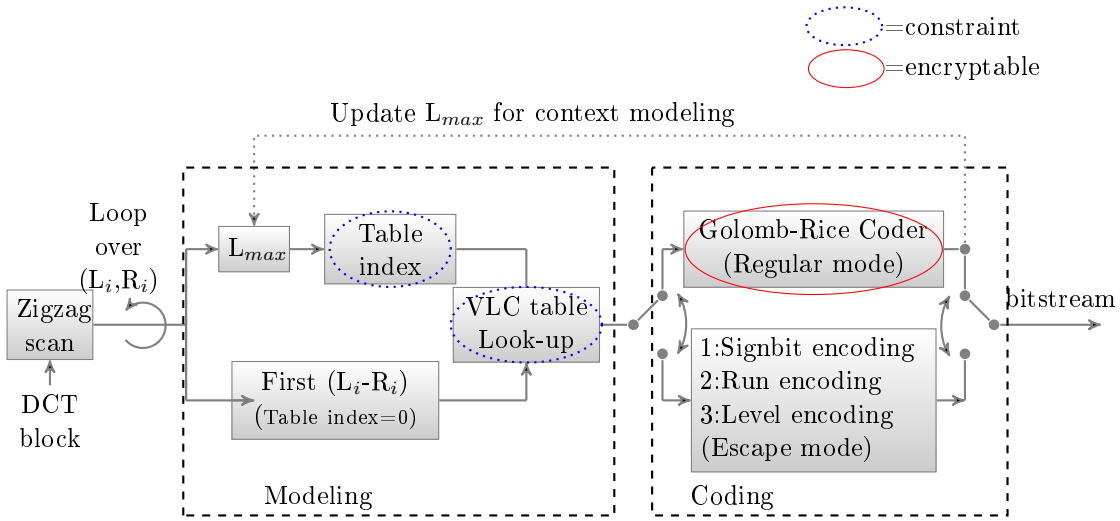


Figure 7.18: Block diagram of C2DVLC showing constraints and encryptable blocks.

Encryption of C2DVLC is not straight forward like that of CAVLC because of these constraints. In CAVLC, code-space is always full and we have specific bits which can be encrypted. But in case of C2DVLC, for *regular mode*, we can encrypt only the levels and their sign bits while taking into account the constraints described above.

As explained in Chapter 4, we map (L_i, R_i) pair to $CodeNumber$, which is then coded using Exp-Golomb codeword, either in *regular* or *escape* mode. In *regular mode*, code-space is not full because of two major **limitations**:

- Non-consecutive $CodeNumbers$ are assigned to consecutive levels.
- We do not have specific bits to be encrypted and the encryption space (ES) is not a power of 2.

In *escape mode*, we can encrypt the sign bit and suffix of the Exp-Golomb codeword. Here code-space is not guaranteed to be full because of second constraint (*i.e.* L_{max} *remains in the same interval*).

Fig. 7.18 encircles the functional blocks with constraints. It also shows the encryptable functional blocks.

To solve the limitations that Non-consecutive $CodeNumbers$ are assigned to consecutive levels, C2DVLC codewords can be mapped to indices which are then encrypted. In this case, the encryption operation can be defined as:

$$Y_i = Encrypt(X_i) = T^{-1}[\mathcal{E}(T(X_i)))], \tag{7.7}$$

where $\mathcal{E}(.)$ is a AES encryption function and $T(.)$ represents a bijective mapping between symbols with the same code-length and indices. The goal of this mapping

is to pass indices to subsequent encryption functions. The corresponding decoding process will be:

$$X_i = Decrypt(Y_i) = T^{-1}[\mathcal{D}(T(Y_i)))], \tag{7.8}$$

where $\mathcal{D}(.)$ is a decryption functions corresponding to $\mathcal{E}(.)$.

For SE-C2DVLC, AES algorithm is used in Cipher Feedback (CFB) mode for encryption, similar to SE of H.264/AVC,. In this mode, AES is a stream cipher. Each ciphertext block Y_i is XORed with the incoming plaintext block X_{i+1} before being encrypted with the key k. For the first iteration, Y_0 is substituted by an initialization vector (IV).

For example, let us code current pair $(L_i, R_i) = (6, 0)$ with $TableIndex = 3$ for *intra_luma* mode as shown in Fig. 7.19. By encoding it with *regular mode* of C2DVLC, its *CodeNumber* will be 17 and its Exp-Golomb codeword will take 7 *bits*. Now let us examine the ES available for this (L_i, R_i) pair.

The first constraint is that only L_i can be encrypted in the (L_i, R_i) pair. So the ES consists of the levels which have valid codeword in $TableIndex = 3$ with $R_i = 0$. These are *CodeNumbers* {8, 0, 2, 4, 9, 11, 17, 21, 25, 33, 39, 45, 55} related to levels {0, 1,..., 12}. The second constraint is that the magnitude of the encrypted L_i should be within the interval that creates the same $TableIndex$ for the next (L_i, R_i). From equation (7.1), we see that the current $L_i = 6$ will increase the $TableIndex$ from 3 to 4 for the next (L_i, R_i) pair to be coded. So the encrypted L_i should also be in the same interval i. e. {5, 6, 7}. From *Table* with $TableIndex = 3$, the *CodeNumbers* for these levels are {11, 17, 21} for positive and {12, 18, 22} for negative sign. The third constraint implies that out of these *CodeNumbers*, only those make the ES which have the same Exp-Golomb codeword length as the original level (7 *bits*). Out of the 6 *CodeNumbers* which have been selected in the last step, five *CodeNumbers* have the same length as $(6, 0)$. The only *CodeNumber* whose length is different is 11. So $(6, 0)$ pair has ES of 5 in this example.

In CAVLC, *escape mode* is rarely used. While in C2DVLC, it is very frequently used and it may be difficult to find a block in which all the transform coefficients are coded using *regular mode*. For *regular mode* of C2DVLC, ES ranges from 1 to 25 and ES is up to 2^n for *escape mode*, where n is the number of bits in the suffix of Exp-Golomb codeword, while respecting the second constraint.

7.3.2 Proposed schemes for real-time SE-C2DVLC

The scheme of mapping encryptable suffixes to indices works fine if codewords are encrypted separately. But when we make a plaintext by encryptable bits of codewords, the encrypted codewords may not be valid codewords because of the second constraint that C2DVLC encryptable bits do not use a full code-space. For example, for a codeword with $ES = 20$, five bits will be used for its index, with only first 20 valid values $\{0, 1, 2, ..., 19\}$ and values $\{20, 21, ..., 31\}$ will not be valid. The encrypted index must lie in the valid range *i.e.*, $\{0, 1, 2, ..., 19\}$.

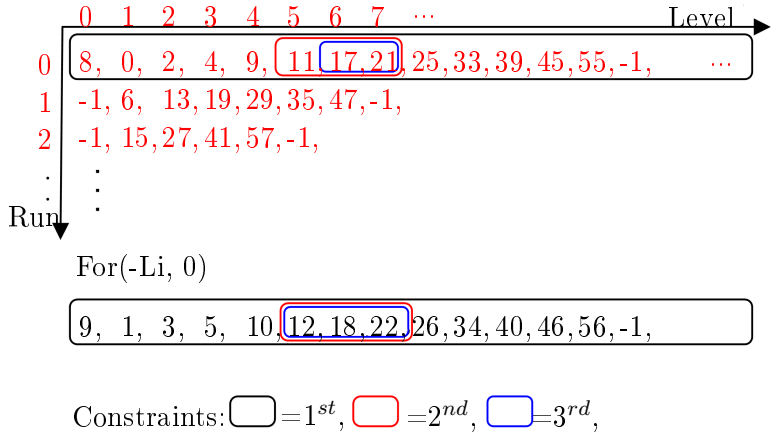


Figure 7.19: $(L_i, R_i) = (6, 0)$ encryption in *regular mode* for $Table$ with $TableIndex = 3$.

We need a scheme to create a plaintext for real-time framework as shown in Fig. 7.20. In this section, we propose two schemes to overcome this limitation for real-time SE of C2DVLC codewords in Section 7.3.2.1 and Section 7.3.2.2. Both of these algorithms are a compromise between the processing power and security of real-time selective encryption.

7.3.2.1 First Approach

First approach targets at utilizing all the available encryption space, at the cost of increased processing power. In this case, we keep track of the available valid ES for indices of codewords in the plaintext X_i. After encryption, we have two types of encrypted indices: valid and non-valid. We replace the valid encrypted indices by the new ones, while the non-valid encrypted indices remain there in the plaintext and are encrypted again to get the valid encrypted indices.

For example, for a 5-bit index with ES = 20 having valid values $\{0, 1, 2, ...19\}$, if the encrypted index lies in first 20 values, it is valid. Otherwise we will encrypt it again till the encrypted index lies in valid range.

This scheme works fine because we know the valid and non-valid values for each index on the decoder side too. If it takes multiple iterations to get a valid encrypted index on the encoder side, it will take exactly the same number of iterations on the decoder side to get the original index, since all the index values between them will not be valid. On the decoder side, we use the same indication of being 'valid' to stop the decryption iterations for a particular index.

7.3.2.2 Second Approach

Second approach aims at saving the processing power, at the cost of a smaller ES. In this approach, if ES is not full (not power of 2), it is decomposed into smaller full code-spaces. Let us suppose, L is the non-full code-space which is to be decomposed

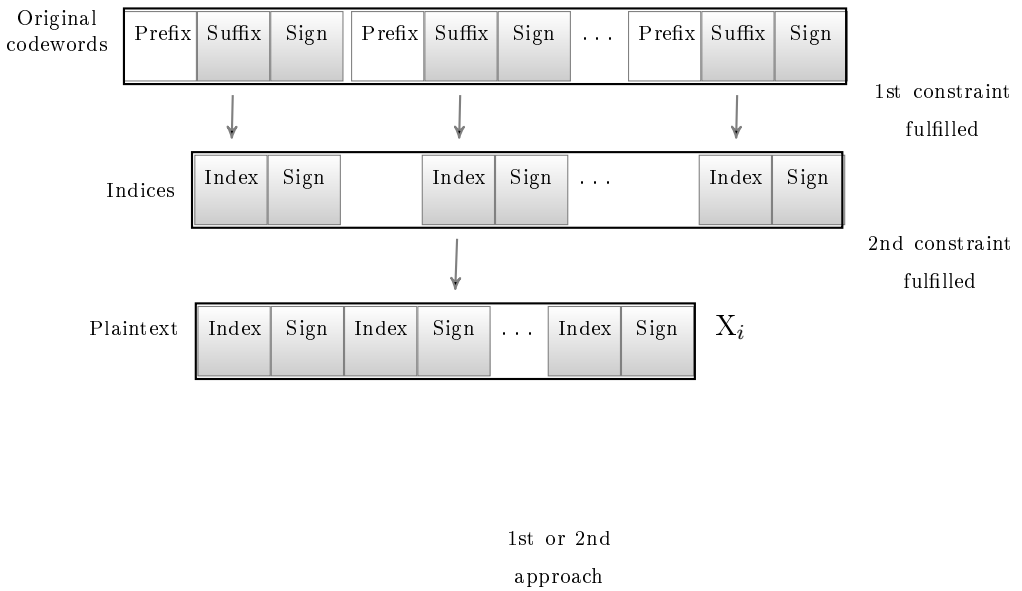
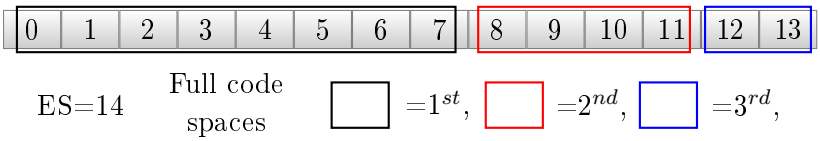


Figure 7.20: Block diagram of C2DVLC showing constraints and encryptable blocks. Suffixes are mapped to indices to fulfill first constraint of non-contiguous *CodeNumbers* for contiguous levels. Second constraint of non-full code-space is handled to have real-time implementation.

to several smaller full code-spaces $l_{full}[n]$. The code-space $l_{full}[i]$ that contains the index value to be encrypted, is the available encryption space in this approach. Let 2^k be the highest power of 2 lesser or equal to L (k is such that $2^k \leq L$), it makes the first encryption space $l_{full}[1]$. This process is repeated on the remaining ES $(L - 2^k)$ recursively to decompose L into full code-spaces (which are power of 2).

For example, a non-full code-space with ES = 14 will be decomposed into three full code-spaces having encryption spaces 8, 4 and 2 respectively as shown in Fig. 7.21. If the index value is 5, its ES will be the first full code-space $\{0, 1, 2, ...7\}$ with $ES = 8$ and its encrypted index will also lie in the same full code-space. Similarly if the index value is 9 or 13, their ESs will be $\{8, 9, .., 11\}$ and $\{12, 13\}$ with $ES = 4$ and $ES = 2$ respectively. Their encrypted index values will also lie in the respective code-spaces as explained in Fig. 7.21. It is important to note that we can have the final full code-space to be $ES = 1$. In that case, the final index value in the ES will not be encrypted. For example, for ES = 15, the code-spaces will be of sizes 8, 4, 2 and 1. In this case, the final index value cannot be encrypted because it lies in code-space with $ES = 1$.

After encryption with AES cipher in the CFB mode, original bits in the bitstream are substituted by the corresponding encrypted bits.



Index	Encryption space (ES)	Encrypted ES
5	1^{st} code-space (0,1,...,7)	1^{st}(0,1,...,7)
9	2^{nd} code-space (8,9,..,11)	2^{nd}(8,9,..,11)
13	3^{rd} code-space (12,13)	3^{rd}(12,13)

Figure 7.21: Example of second real-time approach for SE-C2DVLC.

7.3.3 Experimental results

For the experimental results, nine benchmark video sequences of Appendix A have been used for the analysis in QCIF format.We have presented analyzed of SE-C2DVLC for *intra* sequence and *intra* & *inter* sequence in Section 7.3.3.1 and Section 7.3.3.2 respectively. It also contains analysis of encryption space (ES) and required processing power for first and second SE approaches.

7.3.3.1 *Intra* frames

To demonstrate the efficiency of our proposed scheme for *intra* frames, we have compressed 100 frames of each sequence at 30 fps as *intra*. Table 7.14 compares the PSNR of 100 frames of nine video sequences at QP value of 28 without encryption and with both SE approaches. One can note that the proposed algorithm works well for all types of video sequences having various combinations of motion, texture and objects. Table 7.14 compares the PSNR of 100 frames of nine video sequences at QP value of 28 without encryption and with both SE approaches. One can note that the proposed algorithm works well for all types of video sequences having various combinations of motion, texture and objects. The average PSNR of all the sequences encrypted by first and second approach is 10.2 dB and 10.3 dB for *luma* respectively. Fig. 7.22 shows the encrypted video frames at different QP values for *foreman* for second approach. PSNR comparison over whole range of QP values is given in Table 7.15. One can note that, PSNR of the SE video remains in the same lower range (around 10 dB on average for *luma*) for all QP values.

Table 7.16 demonstrates the encryption space (percentage of bitstream which is encrypted) for both proposed approaches of SE for *intra* frames. For first approach, average ES is 27.65% while it is 25.19% for second approach. Hence to get full code-space, ES of second approach is reduced by 8.9% $((ES1 - ES2)/ES1)$.

7.3.3.2 *Intra* & *inter* frames

For experimental evaluation of *intra* & *inter* frames, *intra period* is set to 10 in a sequence of 100 frames. Table 7.17 verifies the performance of our algorithm for all

Table 7.14: Comparison of PSNR without encryption with first and second real-time SE-C2DVLC approaches for benchmark video sequences at QP = 28 for *intra*.

Seq.	PSNR (Y) (dB)			PSNR (U) (dB)			PSNR (V) (dB)		
	Orig.	1^{st}	2^{nd}	Orig.	1^{st}	2^{nd}	Orig.	1^{st}	2^{nd}
Bus	37.9	7.8	8.5	41.6	26.0	26.4	42.8	27.9	27.9
City	38.1	12.3	12.6	42.9	30.7	30.7	44.2	31.1	31.0
Crew	39.5	10.2	10.3	41.8	25.0	25.2	40.8	22.2	22.4
Football	39.1	11.9	12.0	41.5	16.3	16.1	42.3	24.1	23.8
Foreman	38.9	9.1	8.8	42.1	23.8	24.1	43.9	26.2	26.9
Harbour	37.8	9.8	9.9	42.2	24.4	25.1	43.6	32.5	31.8
Ice	41.4	10.7	10.8	44.5	26.2	25.8	44.8	20.3	19.1
Mobile	37.9	8.7	8.8	38.6	14.5	14.5	38.4	11.8	12.1
Soccer	38.3	11.4	11.3	42.9	22.1	20.8	44.3	24.1	24.4
Avg.	38.8	10.2	10.3	42.0	23.2	23.2	42.8	24.5	24.4

Table 7.15: Comparison of PSNR without encryption and with first and second real-time SE-C2DVLC approaches for *foreman* at different QP values for *intra* .

QP	PSNR (Y) (dB)			PSNR (U) (dB)			PSNR (V) (dB)		
	Orig.	1^{st}	2^{nd}	Orig.	1^{st}	2^{nd}	Orig.	1^{st}	2^{nd}
12	49.6	9.0	8.8	50.1	24.8	24.4	50.8	21.5	21.1
20	44.1	8.9	8.7	45.7	26.3	25.9	47.4	22.1	22.8
28	38.9	9.1	8.8	42.1	23.8	24.1	43.9	26.2	26.9
36	34.4	8.9	9.0	39.3	23.8	23.9	40.2	22.0	21.5
44	30.6	9.1	9.7	37.1	23.9	23.9	37.3	21.7	21.3
52	27.0	10.0	9.8	35.3	25.5	25.1	35.9	20.8	20.1

Table 7.16: Analysis of ES and required processing power with first and second real-time SE-C2DVLC approaches.

Seq.	Encryption Space	
	1^{st} approach (%)	2^{nd} approach (%)
Bus	31.89	28.64
City	27.19	25.46
Crew	20.80	19.80
Football	26.88	24.47
Foreman	24.89	22.91
Harbour	32.10	28.75
Ice	27.27	24.78
Mobile	33.20	28.86
Soccer	24.65	23.02
Avg.	27.65	25.19

Table 7.17: Comparison of PSNR without encryption with first and second real-time SE-C2DVLC approaches for benchmark video sequences at QP value 28 for *intra* & *inter* with *intra period* 10.

	PSNR (Y) (dB)			PSNR (U) (dB)			PSNR (V) (dB)		
Seq.	Orig.	1^{st}	2^{nd}	Orig.	1^{st}	2^{nd}	Orig.	1^{st}	2^{nd}
Bus	36.5	8.0	7.0	41.8	25.2	26.0	43.1	28.0	27.4
City	36.9	12.1	12.3	43.2	31.1	30.4	44.4	31.7	30.9
Crew	38.3	13.4	10.4	42.0	25.4	25.4	40.9	22.4	23.5
Football	37.9	11.8	12.8	41.5	15.2	16.9	42.4	23.4	23.8
Foreman	37.9	8.6	8.2	42.4	25.0	24.4	44.2	26.1	27.2
Harbour	36.2	9.8	9.9	42.4	25.0	28.0	43.9	31.4	33.3
Ice	40.2	10.3	10.8	44.7	26.4	26.1	45.0	18.8	19.8
Mobile	36.1	8.5	9.1	38.8	14.8	12.8	38.5	12.3	11.8
Soccer	37.2	11.5	10.5	43.1	20.4	19.9	44.5	24.2	25.5
Avg.	37.5	10.4	10.1	42.2	23.2	23.3	43.0	24.2	24.8

Table 7.18: Comparison of PSNR without encryption with first and second real-time SE-C2DVLC approaches for *foreman* at different QP values for *intra* & *inter* with *intra period* 10.

	PSNR (Y) (dB)			PSNR (U) (dB)			PSNR (V) (dB)		
QP	Orig.	1^{st}	2^{nd}	Orig.	1^{st}	2^{nd}	Orig.	1^{st}	2^{nd}
12	47.2	9.3	8.7	50.0	25.0	24.7	50.5	23.5	21.2
20	42.8	8.9	8.3	46.0	26.4	27.5	47.7	20.6	23.1
28	37.9	8.6	8.2	42.4	24.9	24.4	44.2	26.1	27.2
36	34.0	8.1	8.7	39.5	23.9	24.9	40.5	21.6	22.3
44	30.4	9.8	8.2	37.3	25.4	23.3	37.7	20.1	23.5
52	27.0	10.7	9.1	35.7	24.4	25.0	36.0	19.8	22.2

video sequences for *intra* & *inter* frames at QP value of 28. Average PSNR of *luma* for all the encrypted sequences is 10.43 *dB*. Results shown in Table 7.18 verify the effectiveness of our scheme over the whole range of QP values for *foreman* for *intra* & *inter* frames.

Table 7.19 demonstrates the compromise for ES and processing power between first and second approaches of SE. For first approach, average ES is 12.93% while it is 12.28% for second approach. Hence to get complexity reduction, we have to reduce our ES by 5%. For encoding process, we need 1.01% more processing power, while it is 0.61% for second approach. For decoding process, we need 5.10% and 3.73% more processing power for first and second SE approaches respectively.

Table 7.19: Analysis of ES and required processing power with first and second real-time SE-C2DVLC approaches.

Seq.	Encryption Space I+P		Processing Power (I+P) encoder		decoder	
	1^{st} (%)	2^{nd} (%)	1^{st} (%)	2^{nd} (%)	1^{st} (%)	2^{nd} (%)
Bus	11.93	11.22	1.38	0.80	5.66	4.04
City	13.38	12.93	1.00	0.62	5.04	3.67
Crew	12.58	12.33	0.62	0.41	3.78	2.66
Football	16.06	14.94	0.82	0.43	5.19	3.91
Foreman	13.61	13.01	0.94	0.57	4.79	3.59
Harbour	12.38	11.72	1.10	0.66	5.48	3.90
Ice	13.07	12.36	0.82	0.46	4.74	3.55
Mobile	11.12	10.28	1.52	1.01	6.50	4.84
Soccer	12.22	11.76	0.88	0.53	4.76	3.44
Avg.	12.93	12.28	1.01	0.61	5.10	3.73

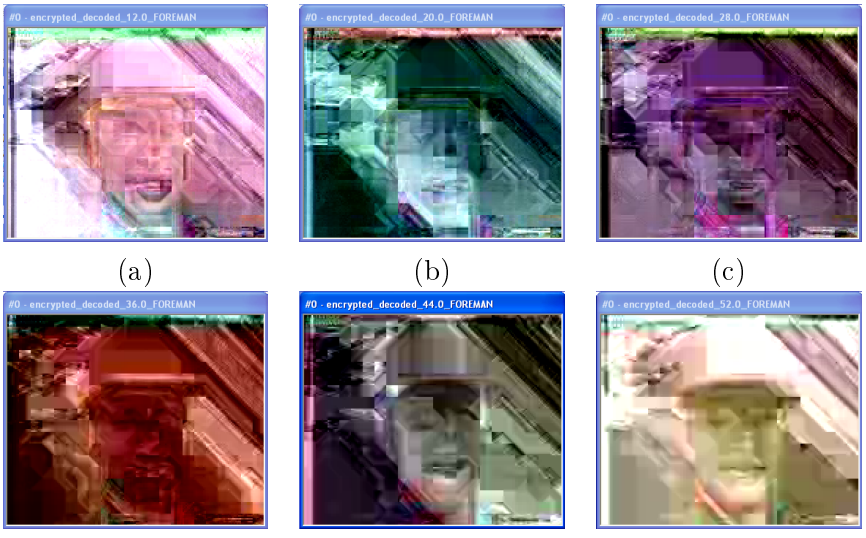


Figure 7.22: Encrypted video using second real-time SE-C2DVLC approach: *foreman* frame # 0 with QP value: (a) 12, (b) 20, (c) 28, (d) 36, (e) 44, (f) 52.

7.3.4 Synopsis

In Section 7.3, a C2DVLC based SE framework for AVS video has been presented. This proposed encryption algorithm is format complaint and keeps the bitrate unchanged. As compared to SE-CAVLC, several additional constraints are fulfilled for creation of plaintext, because code-space of C2DVLC is not full. To cope with this limitation, two real-time schemes have been proposed for its selective encryption.

First approach utilizes all the available ES, while utilizing 0.5% more processing power as compared to second approach, while ES of second approach gets reduced by 5% for (I+P) frames. Second approach is recommended for handheld devices because of lesser requirement of processing power. Since all the constraints posed by the contexts and Exp-Golomb codewords for each NZ, have been fulfilled, encrypted bitstream is fully compliant to AVS format and is decodable by reference AVS decoder.

7.4 Summary

In this chapter, we have presented the real-time selective encryption frameworks for H.264/AVC and AVS. The proposed methods have the advantage of being suitable for streaming over heterogeneous networks because of no change in bitrate. The experiments have shown that we can achieve the desired level of security in each frame, while maintaining the full bitstream compliance, under a minimal set of computational requirements. For H.264/AVC, the proposed schemes fulfill the real-time constraints for CAVLC [Shahid 2009c] and CABAC [Shahid 2009d]. The presented security analysis confirms a sufficient security level for multimedia applications in the context of selective encryption [Shahid 2010a]. For AVS, the proposed encryption algorithm is format complaint and keeps the bitrate unchanged [Shahid 2010c], but creation of plaintext from encryptable bits for real-time implementation is not straight forward, because code-space of C2DVLC is not full. To cope with this limitation, two real-time schemes have been proposed for its selective encryption ,which provides a trade-off between encryption space and processing power [Shahid 2010b]. The proposed system can be extended for ROI specific video protection [Rodrigues 2006] for video surveillance and can be applied to medical video transmission [Puech 2004].

Chapter 8

Selective Encryption Issues for Scalable Video Coding

Contents

8.1 Introduction

Avec l'évolution d'Internet et des réseaux hétérogènes tant en termes de puissance de traitement qu'en termes de largeur de bande de réseau, des utilisateurs différents exigent des versions différentes du même contenu. Cela a donné naissance à la hiérarchisation de contenus vidéo où un seul flux de données contient des versions multiples du même contenu vidéo qui peuvent être différentes en termes de résolution, de débit ou de qualité. Comme des clients différents achètent des versions différentes du même contenu, la préoccupation de la protection et de l'authentification du flux de données hiérarchiques a émergé. Nous avons travaillé sur les architectures hiérarchiques des sousbandes/ondelette (DWTSB) proposée par Hsiang [Hsiang 2008]. Dans cette architecture, une image vidéo est transformée en sous-bandes DWT. La sous-bande LL est codée comme la couche de base, tandis que les sous-bandes haute fréquence sont codées comme des couches ultérieures comme expliqué dans le Chapitre 2. Dans ce chapitre, notre première contribution concerne

l'amélioration du taux de compression des sous-bandes DWTSB dans des couches d'amélioration de flux de données hiérarchiques. Une nouvelle méthodologie de parcours adaptatif pour les images intra encadre la structure de codage hiérarchique basée sur le codage par sousbande/ondelette (DWTSB) est présentée pour le codage hiérarchique de vidéos H.264/AVC (SVC). Il profite de la connaissance antérieure des fréquences qui sont présentes dans les différentes sous-bandes de fréquence plus hautes. Ainsi, juste en modifiant de l'ordre de parcours des images intra, la structure de codage hiérarchique du codeur H.264/AVC, nous pouvons améliorer la compression, sans aucun compromis sur le PSNR. Notre deuxième contribution dans ce chapitre, concerne un nouveau cryptage sélectif et hiérarchique pour les images intra basée sur une structure dyadique de codage hiérarchique basée sur l'ondelette/sousbande (DWTSB) pour H.264/AVC. Il a été réalisé en mélangeant les coefficients transformés quantifiés (QTCs) dans toutes les sous-bandes de DWTSB. Pour rendre ce cryptage hiérarchique, il profite de la connaissance antérieure des fréquences qui sont dominantes dans les différentes sous-bandes des hautes fréquences. Des contraintes en temps réel ont été accomplies avec succès.

With the evolution of Internet to heterogeneous networks both in terms of processing power and network bandwidth, different users demand the different versions of the same content. This has given birth to the scalable era of video content where a single bitstream contains multiple versions of the same video content which can be different in terms of resolutions, frame rates or quality. As different customers purchase different versions of the same content, the concern about the protection and authentication of scalable bitstreams has surfaced. We have worked on subband/wavelet(DWTSB) scalable architecture proposed by Hsiang [Hsiang 2008]. In this architecture, a video frame is transformed to DWT subbands. LL subband is encoded as the base layer while the high frequency subbands are encoded as subsequent layers as explained in Chapter 2. In this chapter, our first contribution is the improvement of the compression ratio of DWTSB subband in enhancement layers (ELs) of a scalable bitstream. A new adaptive scanning methodology for *intra* frame scalable coding framework based on a subband/wavelet(DWTSB) coding approach is presented for H.264/AVC scalable video coding (SVC). It takes advantage of the prior knowledge of the frequencies which are present in different higher frequency subbands. Thus, by just modification of the scan order of the *intra* frame scalable coding framework of H.264/AVC, we can get better compression, without any compromise on PSNR. Our second contribution in this chapter, is a new selective and scalable encryption (SSE) method for *intra* dyadic scalable coding framework based on wavelet/subband (DWTSB) for H.264/AVC. It has been achieved through the scrambling of quantized transform coefficients (QTCs) in all the subbands of DWTSB. To make the encryption scalable, it takes advantage of the prior knowledge of the frequencies which are dominant in different high frequency (HF) subbands. Real-time constraints have been fulfilled successfully.

The rest of the chapter is arranged as follows. We have presented an adaptive scan algorithm for enhancement layers(ELs) of DWTSB based scalable architecture

in Section 8.2. Section 8.3 contains a selective encryption schemes for protection of ELs in a scalable bitstream. Concluding remarks regarding the whole chapter are presented in Section 8.4.

8.2 Adaptive scan for high frequency (HF) subbands in SVC

This section presents a new scanning methodology for intra frame scalable coding framework based on a subband/wavelet (DWTSB) scalable architecture. It takes advantage of the prior knowledge of the frequencies which are present in different higher frequency (HF) subbands. Dyadic intra frame coding method with adaptive scan (DWTSB-AS) performed better for HF subbands as traditional zigzag scan is designed for video content containing most of its energy in low frequencies. Hence, we can get better compression by just modification of the scan order of DCT coefficients. The proposed algorithm has been theoretically justified and is thoroughly evaluated against the current SVC test model JSVM and DWTSB through extensive coding experiments. The simulation results show the proposed scanning algorithm consistently outperforms JSVM and DWTSB in terms of PSNR.

8.2.1 Scan methodology

Let the QTCs be 2-dimensional array given as:

$$P_{m\times n} = \{p(i,j) : 1 \leq i \leq m, i \leq j \leq n\}. \tag{8.1}$$

After scanning the 2-dimensional array, we get a set:

$$Q_{mn} = \{1, ..., mn\}. \tag{8.2}$$

One can note that scanning is a bijective function from $P_{m\times n}$ to Q_{mn}. Indeed, scanning of a 2D array is a permutation in which each element of the array is accessed exactly once.

Natural images generally consist of slow varying areas and contain lower frequencies both horizontally and vertically. After a transformation in the frequency domain, there are lot of non-zero transform coefficients (NZs) in the top left corner. Consequently, zigzag scan is more appropriate than other scans to convert a 2D array of transform coefficients to one directional array.

Entropy coding engine is designed to perform better when:

1. It gets most of the non-zero QTCs in the beginning of scanned and long trail of zeros at its end.

2. Magnitude of non-zero coefficients is higher at the start of the scanned array.

This is the case for slowly changing video data when quantized coefficients are scanned by traditional zigzag scan.

Substituting the image by its wavelet subbands, each subband contains a certain range of frequencies. Traditional zigzag scan is not efficient for all the subbands as the energy is not concentrated in top left corner of 4x4 transform block. Each subband should be scanned in a manner that entropy coding module do maximum possible compression. In other words, most of the non-zero QTCs should be in the beginning and a long trail of zeros at the end of the scanned array.

8.2.2 Analysis of each subband in transform domain

In DWTSB, an image is transformed to wavelet subbands and the LL subband is encoded as base layer by traditional H.264/AVC. In the enhancement layer, LL subband is predicted from the reconstructed base layer. Each high-frequency subband is encoded independently. They are transformed, quantized, scanned and then entropy coded as shown in Fig. 8.1.

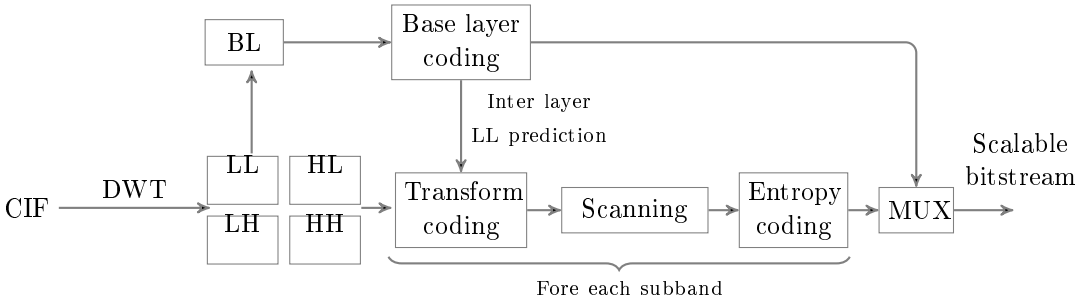


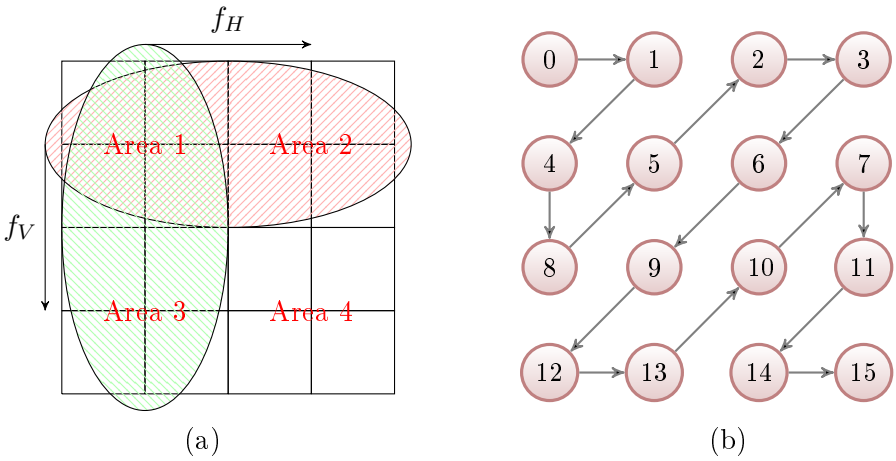
Figure 8.1: DWTSB scalable architecture based on H.264/AVC.

For this work, we have used wavelet critical sampling setting. Daubechies 9/7 wavelet filter set has been used to transform the image to the wavelet subbands at an enhancement layer. The work has been done on 'JVT-W097' [Hsiang 2007] which is referenced H.264 JSVM 8.9 with wavelet framework integrated.

In order to analyze each subband in transform domain, we propose to divide the 2D transform space into 4 areas, *e.g.* as shown in Fig. 8.2.a for LL subband. The area-1 contains most of the energy and has most of NZs. The area-2 and area-3 contain comparatively less number of NZs and only one frequency is dominant in these areas: either horizontal or vertical. The area-4 contains the least number of NZs. Fig. 8.2.a shows the frequency distribution in LL subband. It contains the lower frequencies in both horizontal and vertical directions and transform coefficients in this subband are scanned by traditional zigzag scan as illustrated in Fig. 8.2.b.

8.2.3 Adaptive scan for HF subbands

In this section we propose the adaptive scans for HF subbands. We analyze the frequencies present in HL, LH and HH subbands in order to adapt the scanning processes.



(a) (b)

Figure 8.2: Analysis of LL subband: (a) Dominant frequencies in transformed coefficients of LL subband, (b) Zigzag scan is suitable for such type of frequency distribution.

HL and LH subbands do not contain horizontal and vertical frequencies in equal proportion. HL subband contains most of the high frequencies in horizontal direction while LH contains most of high frequencies in vertical direction. Because of non-symmetric nature of frequencies the scan pattern in not symmetric for HL and LH subbands except in the area-1 which contains both of the frequencies.

In HL subband, there are high horizontal frequencies and low frequencies in vertical direction. Area which contains many NZs should be then in top right corner, as illustrated in Fig. 8.3.a. Based on this, it should be scanned from top right corner to bottom left corner in a natural zigzag, as shown in Fig. 8.3.b. But separation of frequencies in subbands is not ideal and depends on the type of wavelet/subband filter used. It is also affected by rounding errors. So this simple zigzag scan is modified to get better results. Experimental results show that DC coefficient still contains higher energy than other coefficients and should be scanned first. It is followed by a scan from the top left corner in a horizontal fashion till element 11, as illustrated in Fig. 8.3.c. At this position, we have two candidates to be scanned next: element 5 and element 15. We have already scanned the area-1 and zigzag scan is no more feasible. So, element 15 is then selected to be scanned first as it contains higher horizontal frequencies which are dominant in this subband. The same principle is true for the rest of scan lines and unidirectional scan from bottom to top gives better results, thus giving priority to the coefficients which contain higher horizontal frequencies.

Similarly for LH subband, there are low horizontal frequencies and high frequencies in vertical direction. This subband contains most of the NZs in bottom left corner, as illustrated in Fig. 8.4.a. Based on this, LH subband should be scanned in a zigzag fashion from bottom left corner to top right corner as shown in Fig. 8.4.b.

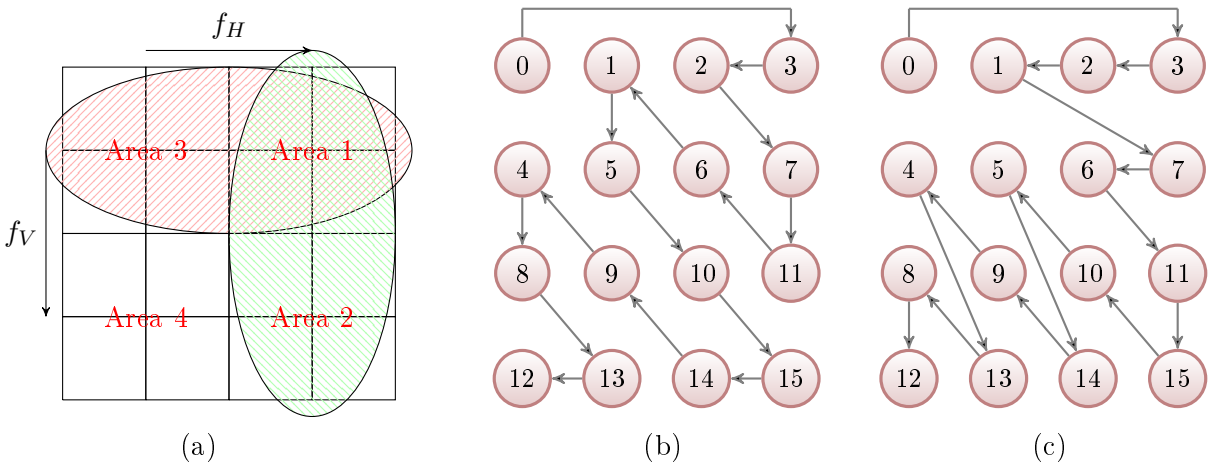


Figure 8.3: Analysis of HL subband: (a) Dominant frequencies in QTCs of this subband, (b) Simple zigzag scan proposed for such type of frequency distribution, (c) Proposed scan for HL subband.

But due to reasons similar to HL subband, this simple scan is modified to a more efficient scan, illustrated Fig. 8.4.c. DC coefficient is scanned first. It is followed by a scan from the bottom left corner in a zigzag fashion till element 5. Thereafter, unidirectional scan which gives priority to the coefficients containing higher vertical frequencies, is performed.

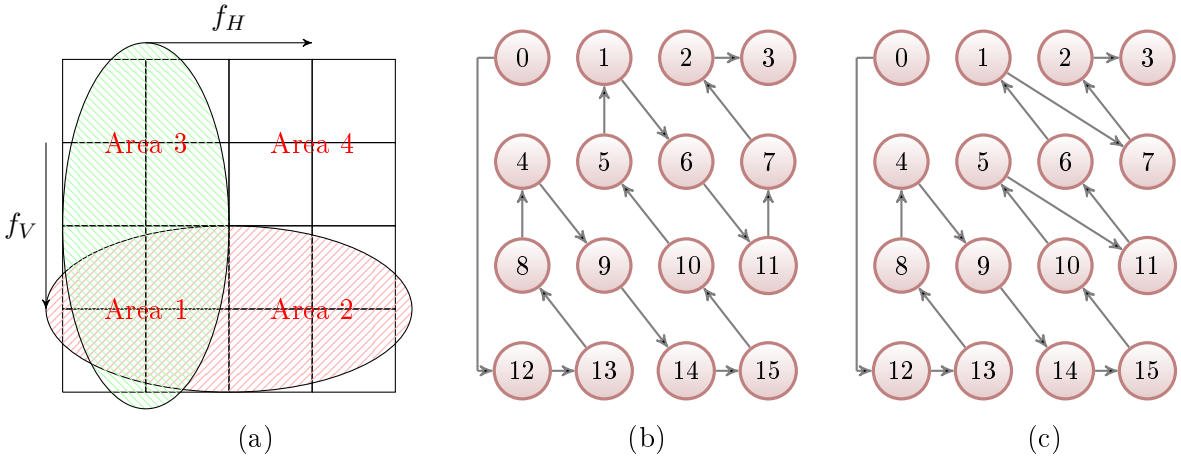


Figure 8.4: Analysis of LH subband: (a) Dominant frequencies in QTCs of this subband, (b) Simple zigzag scan proposed for such type of frequency distribution, (c) Proposed scan for LH subband.

HH subband contains higher frequencies both in horizontal and vertical directions as shown in 8.5.a. Frequencies which contain NZs should then be in bottom

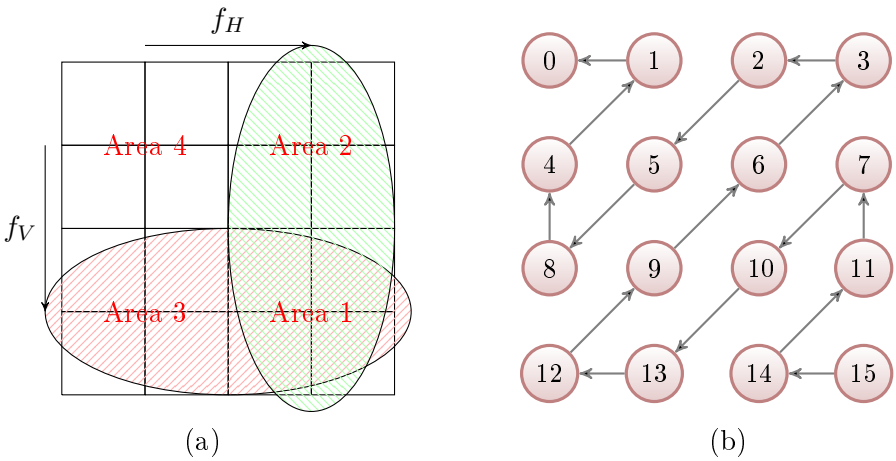


Figure 8.5: Analysis of HH subband: (a) Dominant frequencies in QTCs of this subband, (b) Inverse zigzag scan proposed for such type of frequency distribution.

right. In this subband, DC coefficient contains the least energy and is scanned at the end. So it should be scanned from bottom right corner to top left corner in a zigzag fashion as shown in Fig. 8.5.b.

8.2.4 Experimental results

For the experimental results, nine standard video sequences have been used for the analysis in CIF and QCIF format. To apply our approach we have compressed 150 frames of each sequence at 30 fps.

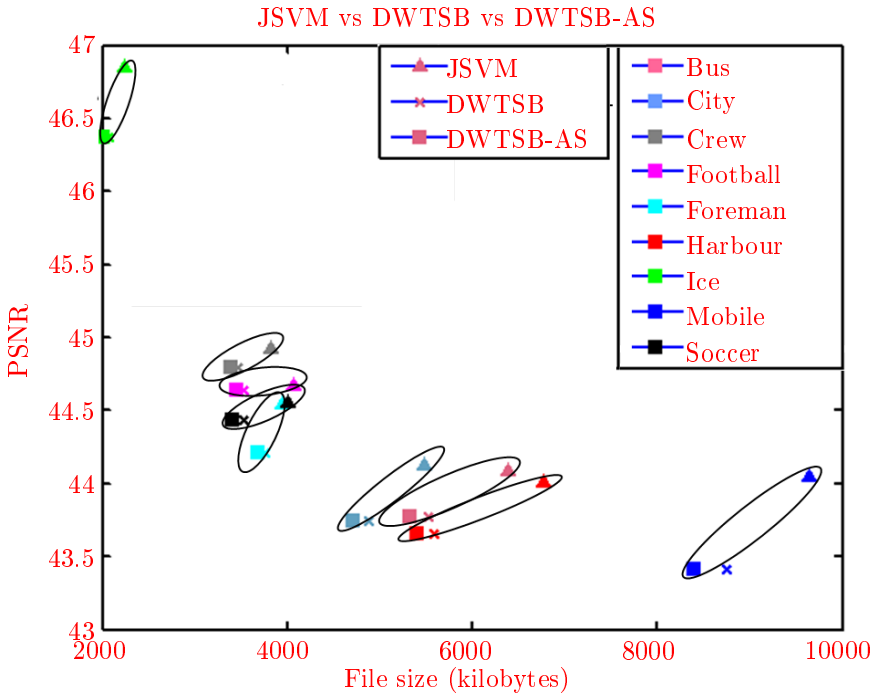
DWTSB dyadic intra frame coding has already been demonstrated to perform better results than JSVM. Results illustrated in Fig. 8.5 for QP value 18 show that DWTSB-AS coding improves results comparing to DWTSB coding. In particular, adaptive scanning helps the entropy coder to perform a better coding and then gives a better compression without any compromise on quality. HH subband offers the best results since the appropriate scan for this subband is exactly opposite to simple zigzag scan. For example, for 'bus' video sequence, DWTSB-AS has reduced the overall bitstream size for the three high frequency subbands (HL, LH and HH) from **2049 kB to 1863 kB** as shown in Fig. 8.5.a. File size of base layer and its residual remains the same since no modification has been made in their scan pattern. The improvements for the overall 2-layer video have been shown in Fig. 8.5.a for all the video sequences. Fig. 8.5.b-d show the file size reduction for HL, LH and HH subbands respectively.

To see the performance as a function of the QP value over the whole rate distortion (R-D) curve, we have tested the proposed scans over 150 frames of the same benchmark video sequences with QP values of 18, 24, 30 and 36. The results show that the performance of adaptive scan is consistent over the whole curve for all the benchmark sequences. Rather adaptive scans performs better at high QP values

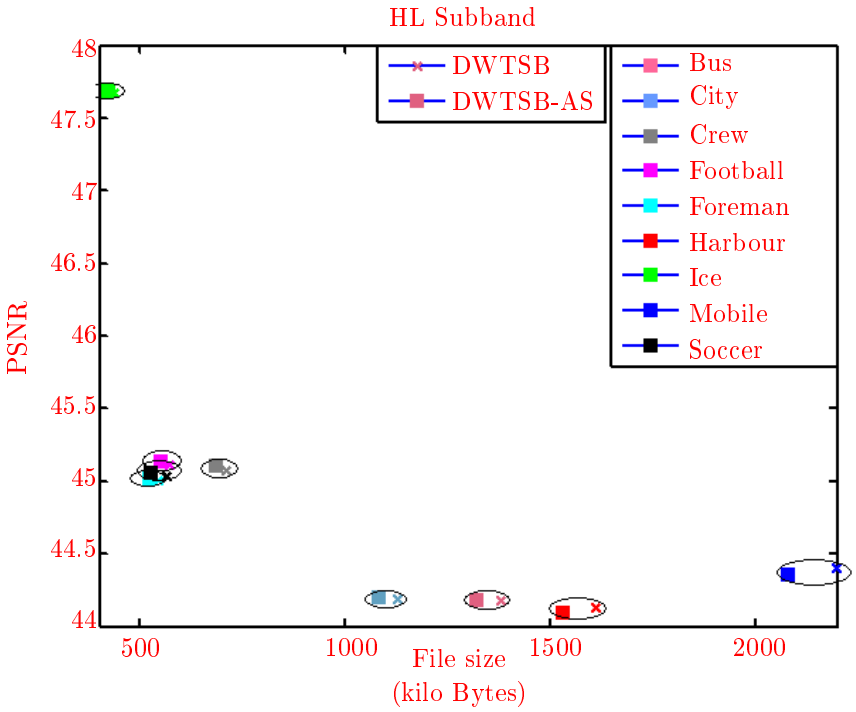
times. Hence our scan performs better for all high frequency subbands over the whole R-D curve. Fig. 8.5.a gives the performance analysis overall 2-layer video *mobile* at different QP values since Fig. 8.5.b-d give the performance analysis for the video *mobile* at different QP values for the three subbands HL, LH and HH respectively.

8.2.5 Synopsis

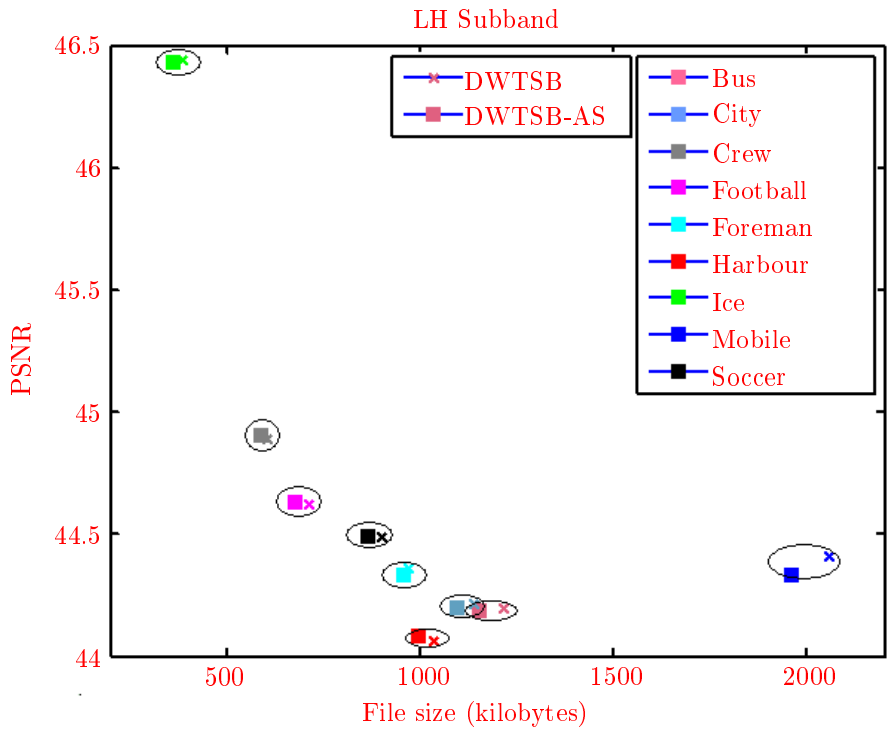
In this section, we have presented a new adaptive scanning methodology for DWTSB scalable architecture of dyadic intra frames. We have described in detail the DWTSB-AS scheme. DWTSB-AS has done a significant file size reduction without any computation load for the same quality as compared to DWTSB coding. Effectiveness of subband-specific scan for DWTSB scalable video has been elaborated by showing experimental results on several benchmark video sequences containing diverse content.


JSVM vs DWTSB vs DWTSB-AS
JSVM
DWTSB
DWTSB-AS
Bus
City
Crew
Football
Foreman
Harbour
Ice
Mobile
Soccer
PSNR
47
46.5
46
45.5
45
44.5
44
43.5
43
2000
4000
6000
8000
10000
File size (kilobytes)

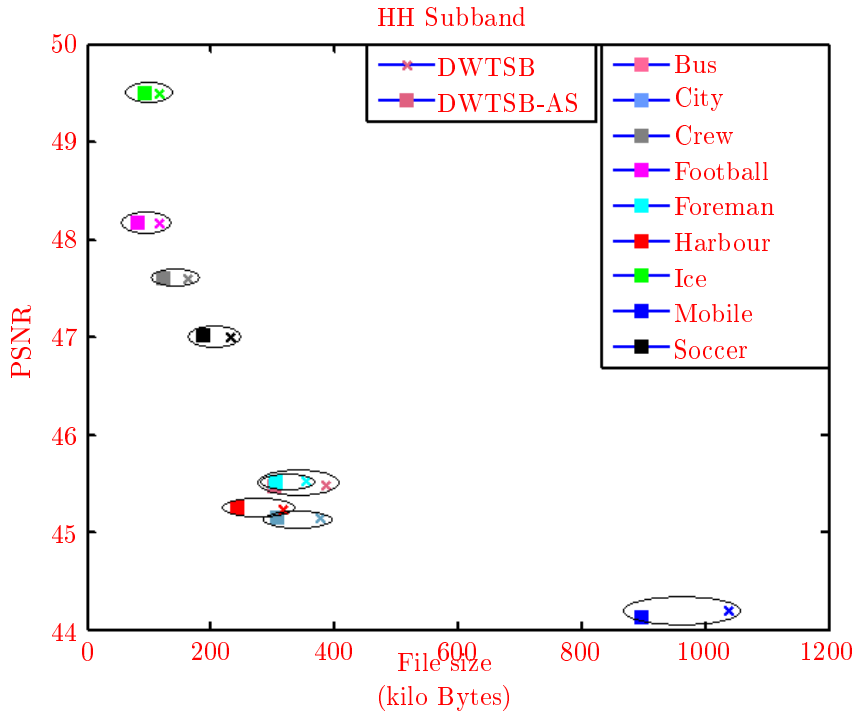
(a)


HL Subband
DWTSB
DWTSB-AS
Bus
City
Crew
Football
Foreman
Harbour
Ice
Mobile
Soccer
PSNR
48
47.5
47
46.5
46
45.5
45
44.5
44
500
1000
1500
2000
File size
(kilo Bytes)

(b)

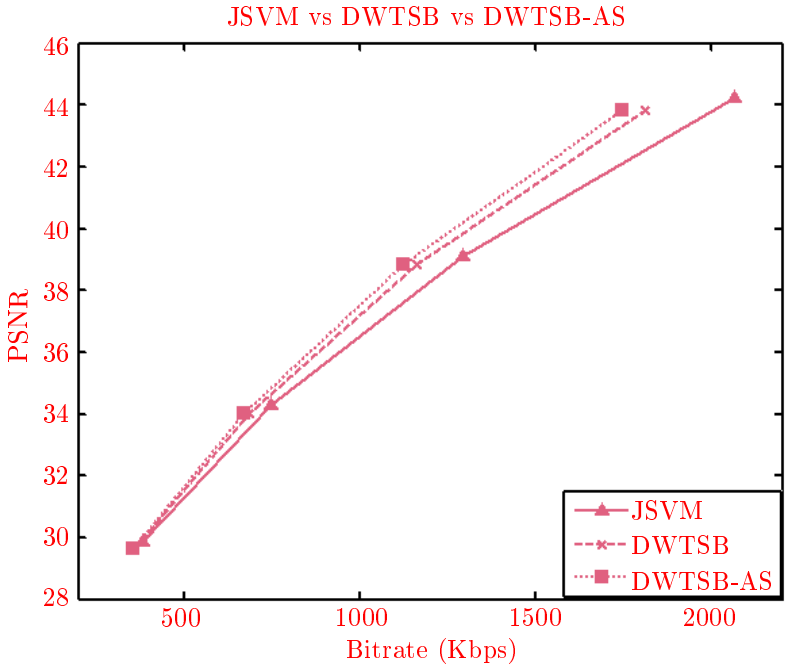


(c)

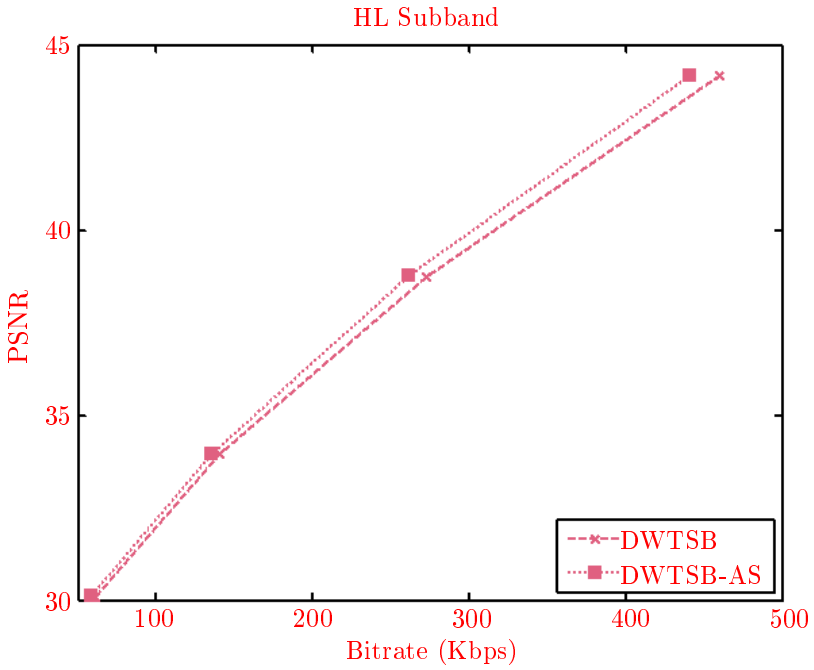


(d)

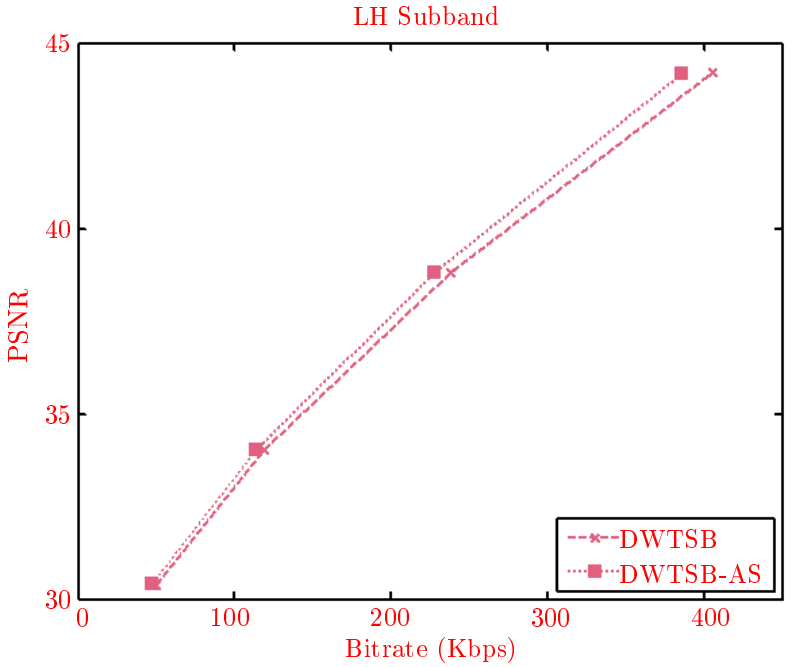
Figure 8.5: Comparison of JSVM, DWTSB and DWTSB-AS: (a) Global comparison for two layer scalable bit streams, (b) HL subband comparison, (c) LH subband comparison, (d) HH subband comparison.


JSVM vs DWTSB vs DWTSB-AS
PSNR
Bitrate (Kbps)
JSVM
DWTSB
DWTSB-AS
46
44
42
40
38
36
34
32
30
28
500
1000
1500
2000

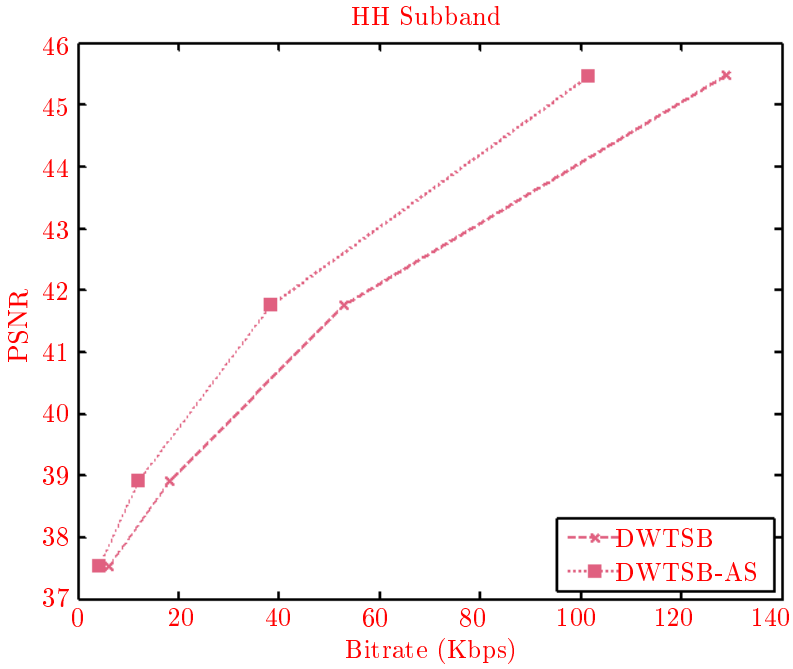
(a)


HL Subband
PSNR
Bitrate (Kbps)
DWTSB
DWTSB-AS
45
40
35
30
100
200
300
400
500

(b)



(c)



(d)

Figure 8.5: Performance comparison of JSVM, DWTSB and DWTSB-AS for *mobile* video sequence over whole QP range: (a) Global comparison for two layer scalable bitstreams, (b) HL subband, (c) LH subband, (d) HH subband.

8.3 Encryption of ELs in DWTSB scalable video

In scalable video, the video bitstream contains a base layer (BL) and number of enhancement layers (ELs). ELs are added to the BL to further enhance resolution, frame rate or quality of video. In spatially scalable bitstream, the resolution of the EL is either equal to or greater than the lower layer. In this section, we present an algorithm for protection of ELs by scrambling of the scan order of QTCs. In Section 8.3.1, we present the proposed algorithm for restricted access of ELs, while the scalability aspects are discussed in Section 8.3.2. Section 8.3.3 contains its performance analysis and experimental results, and concluding remarks are discussed in Section 8.3.4.

8.3.1 Selective encryption method

In DWTSB scalable video, a given image is transformed to wavelet subbands and the LL subband is encoded as base layer by traditional H.264/AVC. In the EL, LL subband is predicted from the reconstructed base layer. Each HF subband is encoded as separate slice. They are transformed, quantized, scanned and then entropy coded. Our main contribution is to scramble the scan order of QTCs in ELs to encrypt them for copyright protection as shown in Fig. 8.6.

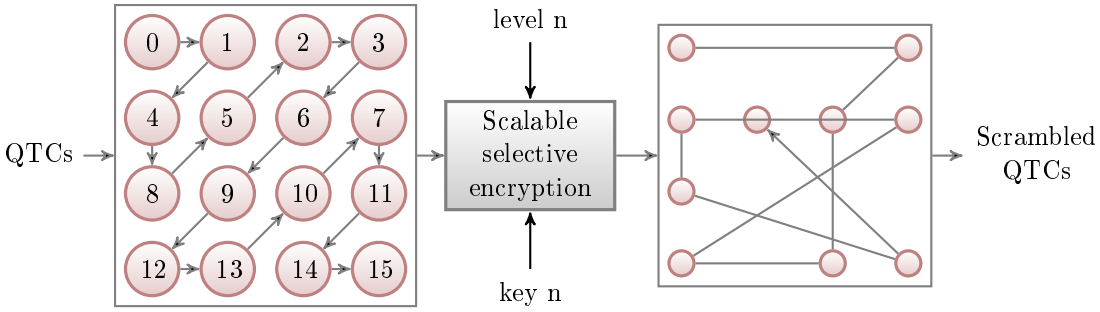


Figure 8.6: Selective encryption of QTCs by scrambling of scan order.

We initialize a pseudo-random number generator (PRNG) with a secret *key*. BL is encoded without encryption. A secret *key* for the 1^{st} EL is watermarked in base layer by embedding the bits of the *key* in the least significant bits (LSBs) of QTCs. In the 1^{st} EL, prediction residual is transformed and quantized. In contrast to the HF subbands, the LL subband, which contains only prediction residual from lower layer, does not contain any frequency in dominance. This makes it the best candidate for embedding the *key* for the immediate higher EL. Embedding the *key* of immediate EL makes it possible to use only one *key* for decryption of certain quality level, thus avoiding the need of multiple *keys* for the decryption of higher quality layers. After watermarking of the LL subband, QTCs, in all the four subbands, are encrypted by scrambling. For a scalable bitstream containing n layers, LL subband

in *nth* layer is not watermarked and all the subbands are only encrypted. The block diagram of the proposed method is presented in Fig. 8.7.

From the watermarking point of view, this scheme is quite flexible and we can use any suitable watermarking method to embed the key in QTCs of LL residual subband.

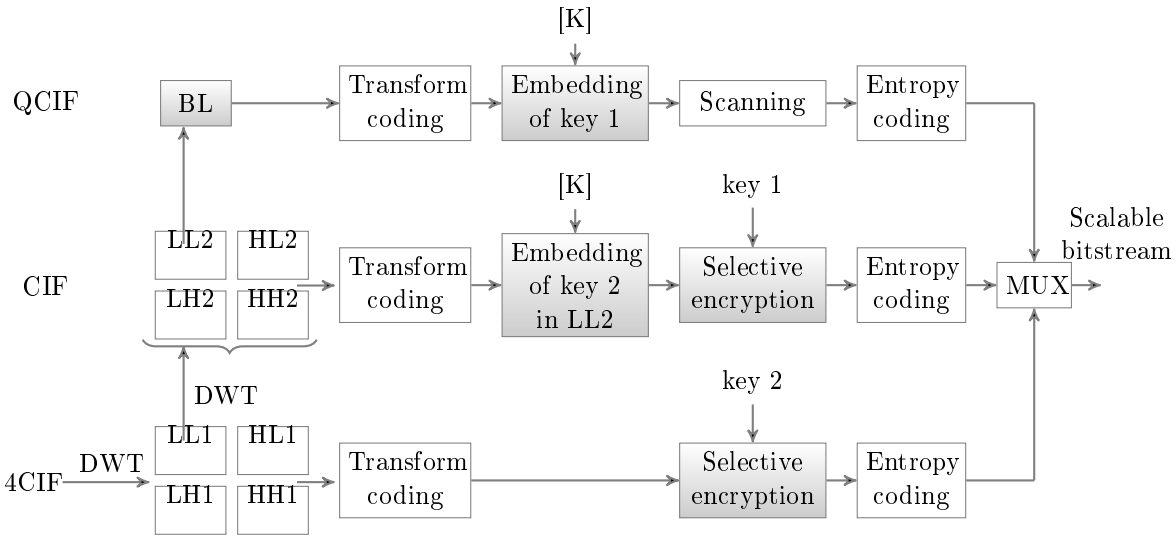
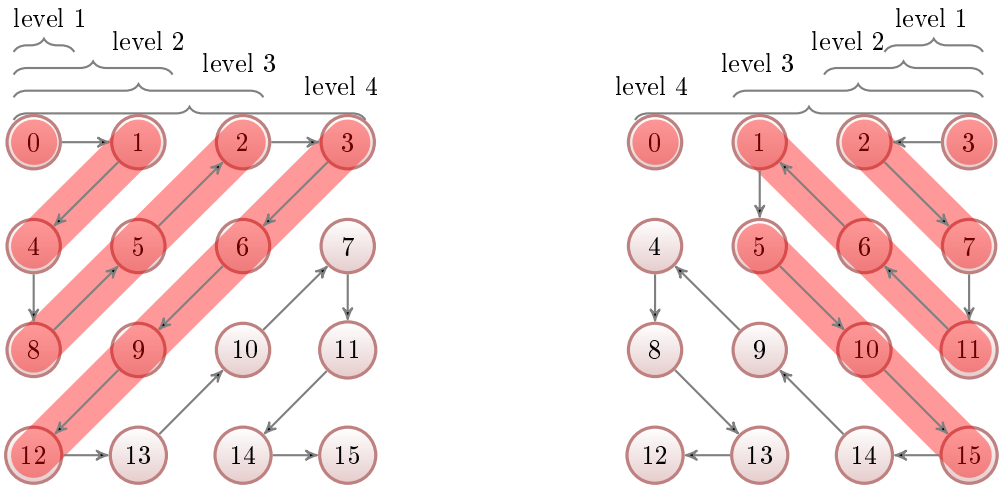


Figure 8.7: Block diagram of selective encryption process of ELs in scalable bitstream.



(a) LL subband (b) HL subband

Figure 8.8: SSE in which selected QTCs are scrambled for each encryption level.

8.3.2 Scalable architecture

To make the SE scheme scalable, we take advantage of a prior knowledge of dominant frequencies in each subband. We have presented five levels of SE. The *level 0* scrambles all the 16 QTCs. It offers the highest security as scrambling space for this level is 16!. The bitrate in this level is approximately equal to that of standard zigzag scan. After that, we leave some QTCs unscrambled, in a diagonal fashion as illustrated in Fig. 8.8.a for LL subband. For *level 1*, we leave the first line unscrambled which reduces the scrambling space to 15!. It assures that the bitrate is always lesser than the original bitstream. *level 2* reduces the scrambling space to 13! by leaving first two lines unscrambled and further reduces the bitrate. For *level* n encryption, we leave first n lines unscrambled. So we reduce the bitrate and the encryption strength for the sake of lesser computations.

It is pertinent to mention that for HF subbands, we leave the coefficients unscrambled according the frequency dominance in that subband as explained in Fig. 8.8.b for HL subband. It is important to note that DC element is also left unscrambled for level 1 for HL and LH subbanbs. For example, we do not scramble QTC at position 15 for *level 1* encryption in HH subband.

8.3.3 Experimental results

For experimental results, five benchmark video sequences of Appendix A with base layer of QCIF resolution and two ELs of CIF and 4CIF resolutions, have been used. We have compressed 150 frames of each sequence as *intra*. Fig. 8.9.a and 8.9.b show a comparison of subframe of 1^{st} frame of 2^{nd} EL (4CIF) of *city* video sequence without encryption and with SSE. All refined detail of video frame including texture and edges, which constitute the EL, has been distorted in the encrypted frame. Frame-wise analysis of *city* is presented in the form of graph in Fig. 8.10.a and 8.10.b at QP value 18. Performance analysis over the whole range of QP values of *city* is given in Table 8.1 for PSNR and bitrate comparison for all the encryption *levels*. Table 8.2 compares the PSNR and the bitrate of all benchmark video sequences at QP value 12 without encryption and with SSE at different encryption *levels*. Our results verifies that this scheme works well for various video contents and over the whole range of QP values.

For all the experimental simulations, 1^{st} EL of CIF resolution has been decrypted properly and it is only the 2^{nd} EL of 4CIF resolution which is left encrypted. If 1^{st} EL is not decrypted, PSNR of encrypted 2^{nd} EL will be very poor (about 10 dB on average).

8.3.4 Synopsis

In Section 8.3, a framework for encryption of ELs of *intra* dyadic scalable *intra* framework for H.264/AVC has been presented. Real-time constraints have been handled successfully by making the encryption scalable and by reducing the bitrate. The experiments have shown that we can achieve the desired level of encryption

Table 8.1: Analysis of trade off among bitrate, PSNR and SE *level* for *city* over whole range of QP values.

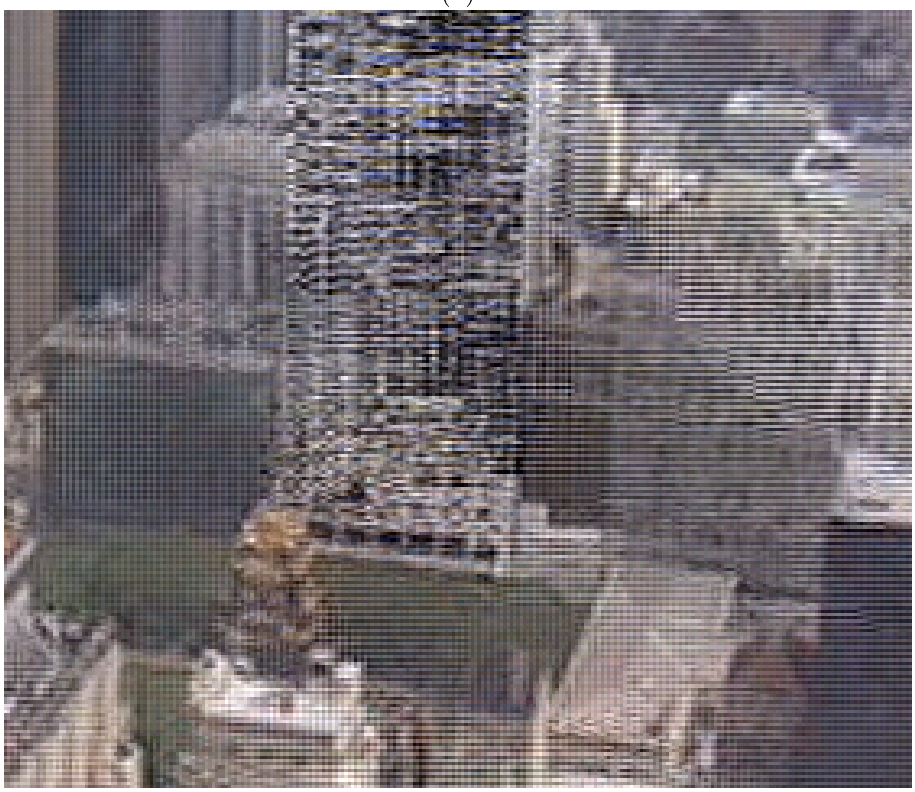
SE level	12.0 (kbps) (dB)	18.0 (kbps) (dB)	24.0 (kbps) (dB)	30.0 (kbps) (dB)	36.0 (kbps) (dB)	42.0 (kbps) (dB)
Orig.	5337 48.71	3386 44.20	1999 39.60	1088 35.19	543 31.17	250 27.60
level 0	5327 19.20	3418 19.75	2045 20.61	1144 20.12	595 18.79	288 18.20
level 1	5285 18.48	3394 19.47	2038 20.24	1147 19.88	598 18.65	286 18.12
level 2	5119 20.79	3270 21.75	1955 22.16	1100 21.04	576 19.28	281 18.87
level 3	5012 22.91	3193 23.99	1910 23.92	1082 22.22	573 20.21	283 19.43
level 4	4942 24.82	3146 25.99	1882 25.73	1071 23.40	571 20.82	283 20.06

Table 8.2: Analysis of change in bitrate and PSNR without encryption and with SSE of benchmark video sequences at QP value 12.

Seq.	Orig. (kbps) (dB)	level 0 (kbps) (dB)	level 1 (kbps) (dB)	level 2 (kbps) (dB)	level 3 (kbps) (dB)	level 4 (kbps) (dB)
City	5337 48.71	5327 19.20	5285 18.48	5119 20.79	5012 22.91	4942 24.82
Crew	4124 49.08	4100 23.16	4066 22.54	3962 25.12	3883 28.04	3831 28.63
Harbour	5260 48.71	5291 19.00	5263 18.57	5080 21.23	4966 23.39	4892 24.67
Ice	2729 49.51	2790 24.48	2772 24.34	2715 26.46	2674 29.13	2649 30.23
Soccer	2348 49.38	2404 21.66	2350 21.19	2282 23.12	2227 26.09	2198 29.04

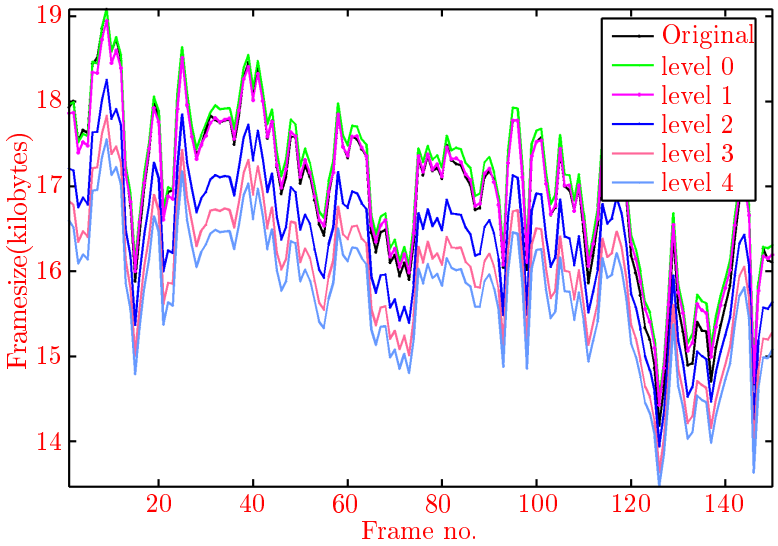
without any considerable escalation in bitrate, if we do not scramble only the first coefficient in the scan pattern. Owing to embedding of *key* in LL subband of subsequent lower layer, a single *key* is required for the decoding of some specific quality level. In future, the proposed scheme can be extended for protection of P and B frames.

(a)

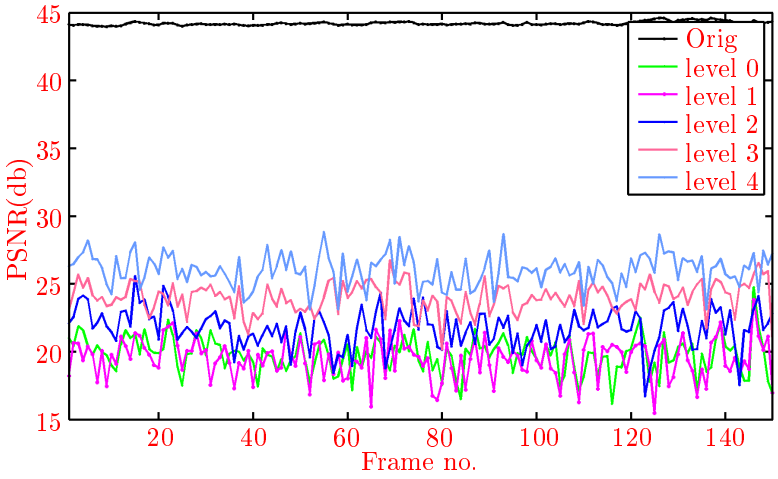


(b)

Figure 8.9: Subframe of 280×240 pixels with offset of (400,200) in original frame of 2^{nd} EL (4CIF) from frame #0 of *city* at QP value 12: (a) Without encryption, (b) With SSE.



(a)



(b)

Figure 8.10: Frame-wise analysis of 2^{nd} EL at 4CIF resolution with QP value 18 of *city*: (a) Frame size, (b) PSNR.

8.4 Summary

In this chapter, we have presented our contribution related to spatially scalable video. First of all, we have presented a new adaptive scanning methodology (DWTSB-AS) for DWTSB scalable coding framework. We have described in detail the DWTSB-AS coding and we have shown that DWTSB-AS coding has done a significant file size reduction without any computation load for the same quality as compared to DWTSB coding [Shahid 2009a]. It is followed by a novel framework for encryption of ELs of *intra* DWTSB architecture. Real-time constraints have been handled successfully by making the encryption scalable and by reducing the bitrate. The experiments have shown that we can achieve the desired level of encryption without any escalation in bitrate [Shahid 2009e].

Chapter 9

Conclusion and Perspective

Dans ce manuscrit, nous avons développé plusieurs méthodes afin de résoudre le problème de protection de vidéos hiérarchiques. D'un coté, nous avons intégré une approche de tatouage dans le module de transformation & quantification et utilisé cette technique pour insérer des empreintes dans le flux vidéo en utilisant les codes de Tardos dans le chapitre 6. D'un autre coté nous avons exploité le module de codage entropique pour développer une méthode de cryptage sélectif qui respecte les contraintes de temps de réel dans le chapitre 7. Finalement, nous nous sommes intéressés à la protection des couches d'amélioration pour des flux de vidéos hiérarchisées dans le chapitre 8. Ceci nous a permis de transformer un codeur vidéo simple en un codeur permettant en même temps d'assurer la compression et la protection d'un flux hiérarchisé. Ensemble, toutes ces méthodes nous permettent de mettre en place des scénarios de protection contre les attaques. Afin de renforcer leur efficacité, nous avons testé et analysé nos approches avec des vidéos de test présentées en annexe A contenant divers contenus et comparé avec d'autres approches de l'état de l'art.

Du fait de l'insertion de la première approche de tatouage dans la boucle de reconstruction, les résultats obtenus ont été très intéressants. Dans le cas du tatouage fragile, la capacité pour la séquence football est de 220.43 *kbps, avec seulement* 1.73 *dB de perte en PSNR et une augmentation de débit de* 3.6 *mbps à* 3.8 *mbps (soit une augmentation de* 6.62 *%) pour des séquences vidéos intra & inter avec une période intra de* 15 *et une valeur de quantification de* 18*. Les résultats obtenus pour l'insertion d'empreintes actives dans des vidéos H.264/AVC ont également été intéressants. Les codes de Tardos, quand ils sont insérés avec une méthode par étalement de spectre dans les vidéos H.264/AVC, montrent qu'ils sont résistants à des attaques négatives qui dégradent le PSNR de* 35 *dB à* 15 *dB avec d'importants artefacts. Au moins la moitié des attaquants sont détectés, même contre cette forte attaque.*

Les méthodes de cryptage sélectif ont bien été intégrées pour les modules de codage entropique CAVLC et CABAC du codeur H.264/AVC. Les PSNR des vidéos sélectivement cryptées sont d'environ 10 *dB en moyenne et leur SSIM moins que* 0.2 *(dans un intervalle de -1 à 1). Le traitement supplémentaire est moins de* 3 *% pour le décodage et moins de* 0.4 *% le codage de séquences en intra & inter avec une quantification de* 18*. Pour AVS, le module C2DVLC l'espace de codage n'est pas disponible et l'implémentation en temps réel n'est pas directement possible. Deux approches temps réel ont été proposées avec un bon compromis entre le temps de traitement et l'espace de cryptage. Une fois que les stratégies ont été mises en place pour un flux*

H.264/AVC, les questions liées à la vidéo hiérarchisées ont été prises en compte. Nous avons proposé une lecture adaptative pour les couches d'amélioration dans l'architecture hiérarchique DWTSB. Les parcours proposés permettent de diminuer la taille de fichier vidéo sans compromis sur la qualité et l'efficacité. Nous avons aussi proposé une méthode de cryptage sélectif hiérarchique pour la protection des couches d'amélioration.

De nombreuses perspectives sont possibles afin d'améliorer ces travaux de protection de vidéos par tatouage et cryptage.

Concernant le tatouage, des masquages perceptuels pour les techniques d'insertion robustes n'ont pas encore été incorporées. Les techniques de tatouage informé accompagné d'un masquage perceptuel doivent améliorer les résultats d'insertion pour le traçage. Dans un futur proche nous visons une méthode de tatouage informé pour H.264/AVC qui pourra être utilisée pour des protections de droits d'auteur et du traçage de traitres.

Pour le tatouage d'empreintes digitales dans des vidéos, la principale limitation des codes Tardos est que la longueur des codes augmente fortement avec le nombre d'utilisateurs. Par exemple, pour 10^6 utilisateurs avec 5 accusés à trouver, la longueur de code est de 8390. Des développements de nouveaux codes de création d'empreintes digitales où l'effort consiste à réduire la longueur des codes tout en conservant une probabilité de faux positifs entre 10^{-6} et 10^{-9} sont envisagés.

Pour le cryptage sélectif hiérarchique, la hiérarchisation temporelle et spatiale sont très similaires et des algorithmes de cryptage sélectif avec une hiérarchisation spatiale pourraient être étendus avec une hiérarchisation temporelle, mais la qualité de hiérarchisation est complètement différente. Le développement d'un cryptage hiérarchique pour un flux avec une qualité hiérarchisée est vraiment intéressant à explorer. L'application sur des régions d'intérêt devrait également être développée dans un contexte de protection de vie privée. Enfin, dans le contexte d'une application complète, la gestion des flux binaires en ligne cryptés hiérarchiquement entre le serveur et les clients est un nouveau problème à prendre au sérieux.

Pour la protection de vidéos, nos travaux sont à la frontière de nombreux domaines, compression vidéo, cryptographie, théorie de l'information, insertion de données cachées et transmission de données. Ce manuscrit est un des premiers à avoir développé des méthodes pour la protection en temps réel sur les codeurs vidéos actuels H.264/AVC et ce travail peut être étendu à d'autres codeurs vidéos comme par exemple AVS, VC1 et VP7.

In this manuscript, we adopted several approaches to deal with the problem of protection of scalable video content. On one hand we integrated our watermark embedding approach in the transform & quantization module and used this technique to fingerprint video bitstream with Tardos fingerprinting code in Chapter 6. While on the other hand we exploited the entropy coding module to perform selective encryption while fulfilling the real-time constraints in Chapter 7. Finally, we addressed the protection of enhancement layers for a scalable video bitstream in Chapter 8.

This helped us to transform a video codec into a monolithic module which performs compression and protection at the same time to generate a scalable protected video bitstream. Together, all these approaches lead to a scenario of putting in place a system that is scalable and offers protection against piracy. To highlight its effectiveness and efficacy, we have analyzed our technique with benchmark video sequences of Appendix A containing diverse contents and compared with other state of the art protection techniques.

Since the presentation of the first approach of watermark embedding inside the reconstruction loop, the results have been appreciating. For fragile watermarking case, the payload for *football* sequence was 220.43 kbps, with only 1.73 dB decrease in PSNR and increase in bitrate from 3.6 mbps to 3.8 mbps (6.62 % increase) for *intra* & *inter* video sequence with *intra period* 15 at QP value 18. The results of active fingerprinting of H.264/AVC video are also appreciating. Tardos fingerprinting code, when embedded with spread spectrum embedding in H.264/AVC video, proved to be resistant to modified negative attack which degraded the PSNR from 35 dB to 15 dB along with ghost artifacts. Half of the colluders were detected even against this strong collusion attack.

Selective encryption performed very well in CAVLC and CABAC entropy coding modules of H.264/AVC. PSNR of selectively encrypted video frames remains about 10 dB on average and SSIM less than 0.2 (on range of -1 to 1). The extra computational load is less than 3 % for decoder and less than 0.4 % for encoder for sequence of *intra* & *inter* frames at QP value of 18. In case of AVS, C2DVLC does not use full code-space and its real-time implementation is not directly possible. Two real-time approaches have been proposed which are good compromise between processing power and encryption space. Once the strategies were put in place for H.264/AVC bitstream, the issues related to scalable video have been focused. An adaptive scan has been proposed for enhancement layers of DWTSB scalable architecture. The proposed scans decrease the file size of video bitstream with no compromise on quality and efficiency. A scalable selective encryption scheme has also been proposed protection of enhancement layers.

Many prospects are there to improve this work regarding watermarking, fingerprinting and encryption.

Regarding watermarking, perceptual masking for robust embedding techniques has not yet been incorporated. Informed watermarking embedding techniques along with psycho visual masking may improve the results for fingerprinting code insertion. In the near future we are mulling the application of informed watermarking techniques for H.264/AVC which can be used for copyright protection and traitor tracing.

Regarding active fingerprinting of video, there are two dimensions for improvement. First, a new fingerprinting code can be designed which is better in terms of code length, detection and threshold selection. For example, for Tardos fingerprinting code, code length increases as the number of users increase. For 10^6 users with 5 accusers to find, the code length will be 8390. There are recent developments of newer fingerprinting codes wherein an effort is being made to reduce the length of

code while keeping the false-positive probability as low as 10^{-6} and 10^{-9}. Second, more effective collusion attacks can be designed by taking into account the information available in the encoded video. In this work, we have not taken into account information available in the encoded video and performed collusion attacks in the spatial domain.

Regarding selective encryption, perceptual mask can be used to encrypt only those coefficients which are important to human visual system (HVS). In this manuscript, we have encrypted only data. It is crucial to investigate format compliant selective encryption of header data in H.264. Moreover, SE of region of interest (ROI) will be important to investigate, due to spatial and temporal prediction of video.

For scalable selective encryption, temporal and spatial scalability are very much similar and SE algorithms for spatial scalability can be extended to temporal scalability, but quality scalability is completely different. Scalable encryption for quality scalable bitstream is really needed to be investigated. The application of ROI should also be investigated in context of privacy protection. Lastly, in the context of a complete application, the online streaming management of encrypted scalable content between the server and its clients must also be given a serious thinking.

For video protection, our work spans a number of fields namely, video compression, cryptography, coding theory, data hiding and video transmission. This manuscript is one of the first elaborated effort for real-time protection of state of the art video codec H.264 and this work can be extended for other video codecs *e.g.*, AVS, VC1 and VP7.

Appendix A

Benchmark Video Sequences

Video content is very diverse in nature. It may contain different combinations of motion (fast/slow, pan/zoom/rotation), color (bright/dull), contrast (high/low) and objects (vehicle, buildings, people). For the experimental results, nine diverse benchmark video sequences have been used for the analysis in QCIF, CIF and **4CIF** resolutions. The details about the sequences are given below:

Name:	**Bus**
Total frames:	150
Resolutions:	QCIF, CIF
Motion:	Smooth, Camera panning
Objects:	Bus, car
Texture:	Texture in background
General comments:	Smooth scene

Name:	**City**
Total frames:	300
Resolutions:	QCIF, CIF, **4CIF**
Motion:	Camera panning, zooming
Objects:	Buildings
Texture:	Texture in background
General comments:	Complex sequence

Name:	**Crew**
Total frames:	300
Resolutions:	QCIF, CIF, **4CIF**
Motion:	Object motion
Objects:	Human (Crew staff)
Texture:	No
General comments:	

Name:	**Football**
Total frames:	260
Resolutions:	QCIF, CIF
Motion:	Camera panning
Objects:	Human (Players)
Texture:	Texture in background
General comments:	Lot of Motion

Name:	**Foreman**
Total frames:	300
Resolutions:	QCIF, CIF
Motion:	Smooth, Camera panning
Objects:	Human face
Texture:	Not much
General comments:	Still background

Name:	**Harbour**
Total frames:	300
Resolutions:	QCIF, CIF, **4CIF**
Motion:	Motion in background
Objects:	Buildings
Texture:	Not much
General comments:	High brightness

Name:	**Ice**
Total frames:	240
Resolutions:	QCIF, CIF, **4CIF**
Motion:	Object motion
Objects:	Human (Human (Players)
Texture:	No
General comments:	High brightness

Name:	**Mobile**
Total frames:	300
Resolutions:	QCIF, CIF
Motion:	Complex motion
Objects:	Several objects)
Texture:	Texture in background
General comments:	Lot of colors

Name:	**Soccer**
Total frames:	300
Resolutions:	QCIF, CIF, **4CIF**
Motion:	Camera panning
Objects:	Human (Players)
Texture:	Texture in background
General comments:	Lot of Motion

Appendix B

List of Publications

Journals

[1] Z. Shahid, M. Chaumont, and W. Puech. Fast Protection of H.264/AVC by Selective Encryption of CAVLC and CABAC for I & P Frames. accepted for publication in IEEE Transactions on Circuits and Systems for Video Technology, Dec 2009.

[2] Z. Shahid, M. Chaumont, and W. Puech. Considering the Reconstruction Loop for Data Hiding of Intra and Inter Frames of H.264/AVC . submitted to Signal, Image and Video Processing, April 2010.

[3] Z. Shahid, M. Chaumont, and W. Puech. Spread Spectrum-Based Watermarking for Tardos Code-Based Fingerprinting for H.264/AVC Video. submitted to EURASIP Journal on Image and Video Processing, September 2010.

Book Chapter

[1] Z. Shahid, M. Chaumont, and W. Puech. *Video Coding, 978-953-7619-X-X*, chapter Scalable Video Coding. In-Tech, 2010. (to appear).

Conferences

[1] Z. Shahid, M. Chaumont, and W. Puech. An Adaptive Scan of High Frequency Subbands of Dyadic Intra Frame in MPEG4-AVC/H.264 Scalable Video Coding. In *Proc. SPIE, Electronic Imaging, Visual Communications and Image Processing*, volume 7257, page 9, San Jose, CA, USA, 2009.

[2] Z. Shahid, M. Chaumont, and W. Puech. Considering the Reconstruction Loop for Watermarking of Intra and Inter Frames of H.264/AVC. In *Proc. European Signal Processing Conference (EUSIPCO'09)*, pages 1794–1798, Glasgow, Scotland, 2009.

[3] Z. Shahid, M. Chaumont, and W. Puech. Fast Protection of H.264/AVC by Selective Encryption. In *Proc. SinFra 2009, Singaporean-French IPAL Symposium, Fusionopolis*, pages –11, Singapore, 2009.

[4] Z. Shahid, M. Chaumont, and W. Puech. Fast Protection of H.264/AVC by Selective Encryption of CABAC. In *Proc. IEEE International Conference on Multimedia & Expo (ICME'09)*, pages 1038–1041, New York, NY, USA, 2009.

[5] Z. Shahid, M. Chaumont, and W. Puech. Fast Protection of H.264/AVC by Selective Encryption of CABAC for I & P frames. In *Proc. European Signal Processing Conference (EUSIPCO'09)*, pages 2201–2205, Glasgow, Scotland, 2009.

[6] Z. Shahid, M. Chaumont, and W. Puech. Selective and Scalable Encryption of Enhancement Layers for Dyadic Scalable H.264/AVC by Scrambling of Scan Patterns. In *Proc. IEEE International Conference on Image Processing (ICIP'09)*, pages 1273 – 1276, Cairo, Egypt, 2009.

[7] Z. Shahid, M. Chaumont, and W. Puech. Over the Real-Time Selective Encryption of AVS Video Coding Standard. In *Proc. European Signal Processing Conference (EUSIPCO'10)*, Aalborg, Denmark, 2010.

[8] Z. Shahid, M. Chaumont, and W. Puech. Selective Encryption of C2DVLC of AVS Video Coding Standard for I & P Frames. In *Proc. IEEE International Conference on Multimedia & Expo (ICME'10)*, pages 1655–1660, Singapore, 2010.

[9] Z. Shahid, M. Chaumont, and W. Puech. Spread Spectrum-Based Watermarking for Tardos Code-Based Fingerprinting for H.264/AVC Video. In *IEEE International Conference on Image Processing (ICIP'10)*, Hong Kong, 2010.

Bibliography

[Abomhara 2010] M. Abomhara, O. Zakaria, O. Khalifa, A. Zaiden and B. Zaiden. *Enhancing Selective Encryption for H.264/AVC Using Advanced Encryption Standard.* International Journal of Computer and Electrical Engineering, vol. 2, no. 2, pages 223–229, 2010. 62, 65, 139, 140

[Alon 2004] N. Alon and U. Stav. *New Bounds on Parent-Identifying Codes: the Case of Multiple Parents.* Combinatorics, Probability and Computing, vol. 13, page 795807, 2004. 69

[Amiri 2009] E. Amiri and G. Tardos. *High Rate Fingerprinting Codes and The Fingerprinting Capacity.* In Proc. Annual ACM-SIAM Symposium on Discrete Algorithms, pages 336–345, Philadelphia, PA, USA, 2009. 77

[Anthapadmanabhan 2007] N. Anthapadmanabhan, A. Barg and I. Dumer. *Fingerprinting Capacity Under the Marking Assumption.* In Proc. IEEE International Symposium on Information Theory, pages 706 –710, 2007. 76, 77

[Arena 2000] S. Arena, M. Caramma and R. Lancini. *Digital Watermarking Applied to MPEG-2 Coded Video Sequences Exploiting Space and Frequency Masking.* In Proc. IEEE International Conference on Image Processing, volume 1, pages 438–441, 2000. 48

[Barg 2001] A. Barg, G. Cohen, S. Encheva, G. Kabatiansky and G. Zemor. *A Hypergraph Approach to the Identifying Parent Property: The Case of Multiple Parents.* SIAM Journal on Discrete Mathematics, vol. 14, pages 423–431, 2001. 69

[Barg 2004] A. Barg and G. Kabatiansky. *A Class of I.P.P. Codes with Efficient Identification.* Journal of Complexity, vol. 20, pages 137–147, 2004. 69

[Bender 1996] W. Bender, D. Gruhl, N. Morimoto and A. Lu. *Techniques for Data Hiding.* I.B.M. Systems Journal, vol. 35, no. 3-4, pages 313–336, 1996. 36

[Bjontegaard 2002] G. Bjontegaard and K. Lillevold. *Context-Adaptive VLC Coding of Coefficients.* In JVT Document JVT-C028, Fairfax, VA, May 2002. 19

[Blackburn 2003] S. Blackburn. *An Upper Bound on The Size of A Code with The K-Identifiable Property.* Journal of Combinatorial Theory, vol. 102, pages 179–185, 2003. 69

[Blakley 1985] G. Blakley, C. Meadows and G. Purdy. *Fingerprinting Long Forgiving Messages.* In Proc. CRYPTO, pages 180–189, 1985. 72

[Blayer 2008] O. Blayer and T. Tassa. *Improved Versions of Tardos' Fingerprinting Scheme.* Designs, Codes and Cryptography, vol. 48, no. 1, pages 79–103, 2008. 76

[Boneh 1998] D. Boneh and J. Shaw. *Collusion-Secure Fingerprinting for Digital Data.* IEEE Transactions on Information Theory, vol. 44, no. 5, pages 1897–1905, September 1998. 71, 72

[Carrillo 2009] P. Carrillo, H. Kalva and S. Magliveras. *Compression Independent Reversible Encryption for Privacy in Video Surveillance.* EURASIP Journal on Information Security, vol. 2009, page 13, 2009. 62, 139, 140

[Cérou 2008] F. Cérou, T. Furon and A. Guyader. *Experimental Assessment of the Reliability for Watermarking and Fingerprinting Schemes.* EURASIP Journal on Information Security, pages 1–12, 2008. 74

[Chae 1999] J. Chae. *Robust Techniques for Hiding Data in Images and Video.* PhD thesis, University of California, Santa Barbara, June 1999. 46

[Chandramouli 2001] R. Chandramouli and N. Memon. *Analysis of LSB Based Image Steganography Techniques.* In Proc. IEEE International Conference on Image Processing, volume 3, pages 1019–1022, 2001. 49

[Chen 1999] B. Chen and G. Wornell. *Dither Modulation: A New Approach to Digital Watermarking and Information Embedding.* volume 3657, pages 342–353. SPIE, 1999. 45

[Chen 2001] B. Chen and G. Wornell. *Quantization Index Modulation: A Class of Provably Good Methods for Digital Watermarking and Information Embedding.* IEEE Transactions on Information Theory, vol. 47, no. 4, pages 1423 –1443, May 2001. 45

[Chen 2009] C. Chen, J. Ni and J. Huang. *Temporal Statistic Based Video Watermarking Scheme Robust against Geometric Attacks and Frame Dropping.* In Proc. International Workshop on Digital Watermarking, pages 81–95, Berlin, Heidelberg, 2009. 48

[Cheng 2000] H. Cheng and X. Li. *Partial Encryption of Compressed Images and Videos.* IEEE Transactions on Signal Processing, vol. 48, no. 8, pages 2439–2445, August 2000. 61, 62, 64

[Chor 1994] B. Chor, A. Fiat and M. Naor. *Tracing Traitors.* Lecture Notes in Computer Science, vol. 839, pages 257–270, 1994. 72

[Chor 2000] B. Chor, A. Fiat, M. Naor and B. Pinkas. *Tracing Traitors.* IEEE Transactions on Information Theory, vol. 46, pages 893–910, 2000. 72

[Chung 2005] Y. Chung, P. Wang, X. Chen, C. Bae, A. Otoom and T. Tran. *A Performance Comparison of High Capacity Digital Watermarking Systems.* In KES (1), pages 1193–1198, 2005. 48, 49

[Costa 1983] M. Costa. *Writing on Dirty Paper.* IEEE Transactions on Information Theory, vol. 29, no. 3, pages 439–441, May 1983. 38, 45

[Cox 1997] I. Cox, J. Kilian, F. Leighton and T. Shamoon. *Secure Spread Spectrum Watermarking for Multimedia.* IEEE Transactions on Image Processing, vol. 6, pages 1673–1687, 1997. 40, 47, 48, 49, 112

[Cox 2008] I. Cox, M. Miller and J. Bloom. Digital Watermarking. Morgan Kaufmann Publishers, 2008. 38, 40, 46, 47

[Craver 2004] S. Craver, B. Liu and W. Wolf. *Histo-Cepstral Analysis for Reverse-Engineering Watermarks.* In Proc. Conference on Information Sciences and Systems, pages 824–826, Princeton, NJ, USA, March 2004. 81

[Cross 2002] D. Cross and B. Mobasseri. *Watermarking for Self-Authentication of Compressed Video.* In Proc. IEEE International Conference on Image Processing, volume 2, pages 913–916, September 2002. 48

[Daemen 2002] J. Daemen and V. Rijmen. *AES Proposal: The Rijndael Block Cipher.* Rapport technique, Proton World Int.l, Katholieke Universiteit Leuven, ESAT-COSIC, Belgium, 2002. 57

[Dai 2003] Y. Dai, L. Zhang and Y. Yang. *A New Method of MPEG Video Watermarking Technology.* In Proc. International Conference on Communication Technology, volume 2, pages 1845–1847, April 2003. 47, 48, 49

[Deguillaume 1999] F. Deguillaume, G. Csurka, J. O'Ruanaidh and T. Pun. *Robust 3D DFT Video Watermarking.* volume 3657 SPIE, pages 113–124, 1999. 47, 48, 49

[Droogenbroeck 2002] M. Droogenbroeck and R. Benedett. *Techniques for a Selective Encryption of Uncompressed and Compressed Images.* In Proc. Advanced Concepts for Intelligent Vision Systems, pages 90–97, Ghent, Belgium, September 2002. 62, 63

[Dufaux 2008] F. Dufaux and T. Ebrahimi. *Scrambling for Privacy Protection in Video Surveillance Systems.* IEEE Transactions on Circuits and Systems for Video Technology, vol. 18, no. 8, pages 1168 –1174, August 2008. 62, 64, 139, 140

[Eggers 2003] J. Eggers, R. Bauml, R. Tzschoppe and B. Girod. *Scalar Costa Scheme for Information Embedding.* IEEE Transactions on Signal Processing, vol. 51, no. 4, pages 1003–1019, April 2003. 45

[Ejima 2000] M. Ejima and A. Miyazaki. *A Wavelet-Based Watermarking for Digital Images and Video.* In Proc. IEEE International Conference on Image Processing, volume 3, pages 678–681, 2000. 49

[Ergun 1999] F. Ergun, J. Kilian and R. Kumar. *A Note on the Limits of Collusion-Resistant Watermarks.* Lecture Notes in Computer Science: Advances in Cryptology, vol. 1592, pages 140–149, 1999. 78

[Fan 2004] L. Fan, S. Ma and F. Wu. *Overview of AVS Video Standard.* In Proc. IEEE International Conference on Multimedia and Expo, pages 423–426, Taipei, Taiwan, 2004. 27, 141

[Fisch 2004] M. Fisch, H. Stgner and A. Uhl. *Layered Encryption Techniques for DCT-Coded Visual Data.* In Proc. 12^{th} European Signal Processing Conference, pages 821–824, Vienna, Austria, September 2004. 62, 64

[Fridrich 2003] J. Fridrich and M. Goljan. *Digital Image Steganography using Stochastic Modulation.* In Proc. SPIE, Electronic Imaging, pages 191–202, Santa Clara, CA, USA, January 2003. 39

[Furon 2000] T. Furon and P. Duhamel. *Robustness of Asymmetric Watermarking Technique.* In Proc. IEEE International Conference on Image Processing, volume 3, pages 21–24, Vancouver, Canada, September 2000. 37

[Furon 2008] T. Furon and P. Bas. *Broken Arrows.* EURASIP Journal on Information Security, 2008. 42

[Golikeri 2007] A. Golikeri, P. Nasiopoulos and Z. Wang. *Robust Digital Video Watermarking Scheme for H.264 Advanced Video Coding Standard.* Journal of Electronic Imaging, vol. 16, no. 4, 2007. 48, 50

[Gong 2008] X. Gong and H. Lu. *Towards Fast and Robust Watermarking Scheme for H.264 Video.* In Proc. IEEE International Symposium on Multimedia, pages 649–653, December 2008. 48, 50

[Goyal 2001] V. Goyal. *Theoretical Foundations of Transform Coding.* IEEE Signal Processing Magazine, vol. 18, no. 5, pages 9–21, September 2001. 11

[Grangetto 2006] M. Grangetto, E. Magli and G. Olmo. *Multimedia Selective Encryption by Means of Randomized Arithmetic Coding.* IEEE Transactions on Multimedia, vol. 8, no. 5, pages 905–917, October 2006. 62, 64, 140

[H263 1998] H263. *ITU Telecommunication Standardization Sector of ITU, Video Coding for Low Bitrate Communication.* In ITU-T Recommendation H.263 Version 2, 1998. 30

[H264 2003] H264. *Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 ISO/IEC 14496-10 AVC).* Rapport technique, Joint Video Team (JVT), Doc. JVT-G050, March 2003. 19, 27

[Hartung 1998] F. Hartung and B. Girod. *Watermarking of Uncompressed and Compressed Video.* Signal Processing, no. 66, pages 283–333, 1998. 48, 51

[Hartung 1999] F. Hartung, J. Su and B. Girod. *Spread Spectrum Watermarking: Malicious Attacks and Counterattacks.* In Proc. SPIE: Security and Watermarking of Multimedia Contents, pages 147–158, January 1999. 112

[He 2003] D. He, Q.Sun and Q.Tian. *A Semi-Fragile Object Based Video Authentication System.* In Proc. International Symposium on Circuits and Systems, volume 3, pages 814–817, May 2003. 37

[He 2006] S. He and M. Wu. *Joint Coding and Embedding Techniques for Multimedia Fingerprinting.* IEEE Transactions on Information Forensics and Security, vol. 1, no. 2, pages 231–247, June 2006. 70, 73, 74, 76

[Hollmann 1998] H. Hollmann, J. Lint, J. Linnartz and L. Tolhuizen. *On Codes with the Identifiable Parent Property.* Journal of Combinatorial Theory, vol. 82, pages 121–133, 1998. 69

[Hsiang 2007] S. Hsiang. *CE3: Intra-frame Dyadic Spatial Scalable Coding Based on a Subband/Wavelet Filter Banks Framework.* Joint Video Team, Doc. JVT-W097, April 2007. 154

[Hsiang 2008] S. Hsiang. *A New Subband/Wavelet Framework for AVC/H.264 Intra-Frame Coding and Performance Comparison with Motion-JPEG2000.* In SPIE, Visual Communications and Image Processing, volume 6822, pages 1–12, January 2008. 30, 31, 151, 152

[Huffman 1952] D. Huffman. *A Method for the Construction of Minimum Redundancy Codes.* In Proc. IRE, volume 40, pages 1098–1101, 1952. 11

[Jakimoski 2008] G. Jakimoski and K. Subbalakshmi. *Cryptanalysis of Some Multimedia Encryption Schemes.* IEEE Transactions on Multimedia, vol. 10, no. 3, pages 330–338, April 2008. 64

[Jiangtao 2006] W. Jiangtao, K. Hyungjin and J. Villasenor. *Binary Arithmetic Coding with Key-based Interval Splitting.* IEEE Signal Processing Letters, vol. 13, no. 2, pages 69–72, February 2006. 62, 64, 140

[Kahn 1967] D. Kahn. The Codebreakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet. Scribner, 1967. 54

[Kang 2003] X. Kang, J. Huang, Y. Shi and Y. Lin. *A DWT-DFT Composite Watermarking Scheme Robust to Both Affine Transform and JPEG Compression.* IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 8, pages 776–786, August 2003. 47, 49

[Kapotas 2007] S. Kapotas, E. Varsaki and A. Skodras. *Data Hiding in H.264 Encoded Video Sequences.* In Proc. IEEE International Workshop on Multimedia Signal Processing, October 2007. 48, 50

[Katzenbeisser 2000] S. Katzenbeisser and F. Petitcolas, editeurs. Information Hiding Techniques for Steganography and Digital Watermarking. Artech House, Inc., Norwood, MA, USA, 2000. 49

[Kerckhoffs 1883] A. Kerckhoffs. *La Cryptographie Militaire.* Journal des Sciences Militaires, vol. 9, pages 5–38, 1883. 36, 55

[Kilian 1998] J. Kilian, F. Leighton, L. Matheson, T. Shamoon, R. Tarjan and F. Zane. *Resistance of Digital Watermarks to Collusive Attacks.* In Proc. IEEE International Symposium on Information Theory, August 1998. 78

[Kim 2007] S. M. Kim, S. B. Kim, Y. Hong and C. Won. *Data Hiding on H.264/AVC Compressed Video*, 2007. 48, 50

[Lai 1992] X. Lai. *On the Design and Security of Block Ciphers.* ETH Series in Information Processing, Konstanz: Hartung-Gorre Verlag, vol. 1, 1992. 57

[Langelaar 2001] G. Langelaar and R. Lagendijk. *Optimal Differential Energy Watermarking of DCT Encoded Images and Video.* IEEE Transactions on Image Processing, vol. 10, pages 148–158, 2001. 89

[Lee 1999] P. Lee and M. Chen. *Robust Error Concealment Algorithm for Video Decoder.* IEEE Transactions on Consumer Electronics, vol. 45, no. 3, pages 851–859, August 1999. 37

[Li 2001] W. Li. *Overview of Fine Granularity Scalability in MPEG-4 Video Standard.* IEEE Transactions on Circuits and Systems for Video Technology, vol. 11, no. 3, pages 301–317, March 2001. 61, 140

[Li 2003] K. Li and X. Zhang. *Reliable Adaptive Watermarking Scheme Integrated with JPEG2000.* In Proc. International Symposium on Image and Signal Processing and Analysis, Rome, Italy, September 2003. 37

[Li 2007] Q. Li and I. Cox. *Using Perceptual Models to Improve Fidelity and Provide Resistance to Volumetric Scaling for Quantization Index Modulation Watermarking.* IEEE Transactions on Information Forensics and Security, vol. 2, no. 2, pages 127–139, June 2007. 47, 48

[Li 2008] C. Li, X. Zhou and Y. Zong. *NAL Level Encryption for Scalable Video Coding.* Lecture notes in Computer Science, Springer, no. 5353, pages 496–505, 2008. 62, 65, 139, 140

[Li 2009] J. Li, H. Liu, J. Huang and Y. Zhang. *A Robust Watermarking Scheme for H.264.* Lecture Notes in Computer Science: Digital Watermarking, pages 1–15, 2009. 49

[Lian 2005] S. Lian, Z. Liu, Z. Ren and Z. Wang. *Selective Video Encryption Based on Advanced Video Coding*. Lecture notes in Computer Science, Springer-verlag, no. 3768, pages 281–290, 2005. 62, 63

[Lian 2007] S. Lian, Z. Liu, Z. Ren and H. Wang. *Commutative Encryption and Watermarking in Video Compression*. IEEE Transactions on Circuits and Systems for Video Technology, vol. 17, no. 6, pages 774–778, June 2007. 62, 63, 139, 140

[Lin 1999] E. Lin and E. Delp. *A Review of Data Hiding in Digital Images*. In Proc. The Image Processing, Image Quality, Image Capture Systems Conference, pages 274–278, April 1999. 39

[Lin 2005] E. Lin, A. Eskicioglu, R. Lagendijk and E. Delp. *Advances in Digital Video Content Protection*. Proc. of the IEEE, vol. 93, no. 1, pages 171–183, January 2005. 62

[Lin 2009] W. Lin, S. He and J. Bloom. *Performance Study and Improvement on ECC-Based Binary Anti-Collusion Forensic Code for Multimedia*. In Proc. ACM workshop on Multimedia and security, pages 93–98, New York, NY, USA, 2009. xiii, 73, 75, 76

[Linnartz 1998] J. Linnartz and J. Talstra. *MPEG PTY-Marks: Cheap Detection of Embedded Copyright Data in DVD-video*. In Proc. European Symposium on Research in Computer Security, pages 221–240, 1998. 47

[Lookabaugh 2004] T. Lookabaugh and D. Sicker. *Selective Encryption for Consumer Applications*. IEEE Communications Magazine, vol. 42, no. 5, pages 124–129, May 2004. 61

[Lu 2005] C. Lu, J. Chen and K. Fan. *Real-Time Frame-Dependent Video Watermarking in VLC Domain*. Signal Processing: Image Communication, vol. 20, no. 7, pages 624 – 642, 2005. 48, 50, 89

[Lukac 2005] R. Lukac and K. Plataniotis. *Bit-Level Based Secret Sharing for Image Encryption*. Pattern Recognition, vol. 38, no. 5, pages 767–772, May 2005. 62, 64

[Ma 2006] S. Ma and W. Gao. *Low Complexity Integer Transform and Adaptive Quantization Optimization*. Journal of Computer Science and Technology, vol. 21, no. 3, pages 354–359, 2006. 27

[Malvar 2003] H. Malvar, A. Hallapuro, M. Karczewicz and L. Kerofsky. *Low-Complexity Transform and Quantization in H.264/AVC*. IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 7, pages 598–603, 2003. 23

[Malver 2003] H. Malver and D. Florencio. *Improved Spread Spectrum: A New Modulation Technique for Robust Watermarking.* IEEE Transactions on Signal Processing, vol. 51, pages 898–905, 2003. 41

[Maniccam 2004] S. Maniccam and N. Bourbakis. *Image and Video Encryption using SCAN Patterns.* Pattern Recognition, vol. 37, no. 4, pages 725 –737, 2004. Agent Based Computer Vision. 62, 63

[Marpe 2003] D. Marpe, H. Schwarz and T. Wiegand. *Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard.* IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 7, pages 620–636, July 2003. 20

[Martin 2005] K. Martin, R. Lukac and K. Plataniotis. *Efficient Encryption of Wavelet-Based Coded Color Images.* Pattern Recognition, vol. 38, no. 7, pages 1111–1115, July 2005. 62, 64

[Marvel 1999] L. Marvel, C. Boncelet and C. Retter. *Spread Spectrum Image Steganography.* IEEE Transactions on Image Processing, vol. 8, pages 1075–1083, 1999. 41

[Miller 2004] M. Miller, G. Doerr and I. Cox. *Applying Informed Coding and Embedding to Design a Robust High Capacity Watermark.* IEEE Transactions on Image Processing, vol. 13, no. 2, pages 792–807, June 2004. 48

[Mobasseri 2007] B. Mobasseri and Y. Raikar. *Authentication of H.264 Streams by Direct Watermarking of CAVLC Blocks.* In Proc. Security, Steganography, and Watermarking of Multimedia Contents IX, volume 6505 SPIE, San Jose, CA, 2007. 48, 50

[Moccagatta 2002] I. Moccagatta and K. Ratakonda. *A Performance Comparison of CABAC and VCL-Based Entropy Coders for SD and HD Sequences.* Rapport technique, Joint Video Team (JVT), Doc. JVT-E079r2, October 2002. 24

[Moffat 1998] A. Moffat, R. Neal and I. Witten. *Arithmetic Coding Revisited.* ACM Transactions on Information Systems, vol. 16, no. 3, pages 256–294, 1998. 11

[Moulin 2003] P. Moulin and J. O'Sullivan. *Information-Theoretic Analysis of Information Hiding.* IEEE Transactions on Information Theory, vol. 49, pages 563–593, March 2003. 70, 112

[Moulin 2005] P. Moulin and R. Koetter. *Data-Hiding Codes.* Proc. of the IEEE, pages 2083–2126, 2005. 70

[MPEG2 2000] MPEG2. *ISO/IEC 13818-2:2000 Information Technology – Generic Coding of Moving Pictures and Associated Audio Information: Video, 2nd Edition.* 2000. 30

[MPEG4 2004] MPEG4. *ISO/IEC 14496-2:2004 Information Technology – Coding of Audio-Visual Objects: Visual, 3rd Edition*. 2004. 30, 31

[Mukherjee 2000] D. Mukherjee, J. Chae and S. Mitra. *A Source and Channel-Coding Framework for Vector-Based Data Hiding in Video*. IEEE Transactions on Circuits and Systems for Video Technology, vol. 10, no. 4, pages 630 –645, June 2000. 46

[Noorkami 2005] M. Noorkami and R. Mersereau. *Compressed-Domain Video Watermarking for H.264*. In Proc. IEEE International Conference on Image Processing, pages 890–893, Genoa, Italy, September 2005. 48, 50

[Noorkami 2008] M. Noorkami and R. Mersereau. *Digital Video Watermarking in P-Frames With Controlled Video Bit-Rate Increase*. IEEE Transactions on Information Forensics and Security, vol. 3, no. 3, pages 441–455, September 2008. 48, 50, 111

[Oostveen 2009] J. Oostveen, T. Kalker and J. Haitsma. *Visual Hashing of Digital Video: Applications and Techniques*. In Proc. SPIE, Applications of Digital Image Processing XXIV, volume 4472, pages 121–131, San Diego CA, USA, 2009. 48

[Ou 2006] S. Ou, H. Chung and W. Sung. *Improving the Compression and Encryption of Images using FPGA-Based Cryptosystems*. Multimedia Tools and Applications, vol. 28, no. 1, pages 5–22, January 2006. 62, 64

[Podilchuk 1998] C. Podilchuk and W. Zeng. *Image-Adaptive Watermarking using Visual Models*. IEEE Journal on Selected Areas in Communications, vol. 16, pages 525–539, May 1998. 46

[Pröfrock 2006] D. Pröfrock, M. Schlauweg and E. Müller. *A New Uncompressed-Domain Video Watermarking Approach Robust to H.264/AVC Compression*. In Proc. IASTED International Conference on Signal Processing, Pattern Recognition, and Applications, pages 99–104, Anaheim, CA, USA, 2006. 48, 49

[Puech 2004] W. Puech and J. Rodrigues. *A New Crypto-Watermarking Method for Medical Images Safe Transfer*. In Proc. European Signal Processing Conference, Vienna, Austria, 2004. 150

[Qiu 2004] Gang Qiu, Pina Marziliano, Anthony T. S. Ho, Dajun He and Qibin Sun. *A Hybrid Watermarking Scheme for H.264/AVC Video*. Pattern Recognition, International Conference on, vol. 4, pages 865–869, 2004. 48, 49

[Ramkumar 2000a] M. Ramkumar. *Data Hiding in Multimedia: Theory and Applications*. PhD thesis, New Jersey Institute of Technology, January 2000. 46

[Ramkumar 2000b] M. Ramkumar and A. Akansu. *Robust Protocols for Proving Ownership of Images.* In Proc. International Conference on Information Technology: Coding and Computing, pages 22–27, 2000. 37

[Richardson 2003] I. Richardson. H.264 and MPEG-4 Video Compression. John Wiley & Sons, 2003. 24

[Rodrigues 2006] J. Rodrigues, W. Puech and A. Bors. *Selective Encryption of Human Skin in JPEG Images.* In Proc. IEEE International Conference on Image Processing, pages 1981–1984, Atlanta, USA, October 2006. 62, 150

[Ruanaidh 1998] J. Ruanaidh and T. Pun. *Rotation, Scale and Translation Invariant Spread Spectrum Digital Image Watermarking.* Signal Processing, vol. 66, no. 3, pages 303 – 317, 1998. 41

[Said 2005] A. Said. *Measuring the Strength of Partial Encryption Scheme.* In Proc. IEEE International Conference on Image Processing, volume 2, pages 1126–1129, Genova, Italy, 2005. 61, 138

[Schneier 1995] B. Schneier. Applied cryptography. Wiley, New-York, USA, 1995. 56, 123

[Schwarz 2007] H. Schwarz and T. Wiegand. *Overview of the Scalable Video Coding Extension of the H.264/AVC Standard.* IEEE Transactions on Circuits and Systems for Video Technology, vol. 17(9), pages 1103–1120, September 2007. 30

[Shahid 2009a] Z. Shahid, M. Chaumont and W. Puech. *An Adaptive Scan of High Frequency Subbands of Dyadic Intra Frame in MPEG4-AVC/H.264 Scalable Video Coding.* In Proc. SPIE, Electronic Imaging, Visual Communications and Image Processing, volume 7257, page 9, San Jose, CA, USA, 2009. 169

[Shahid 2009b] Z. Shahid, M. Chaumont and W. Puech. *Considering the Reconstruction Loop for Watermarking of Intra and Inter Frames of H.264/AVC.* In Proc. European Signal Processing Conference, pages 1794–1798, Glasgow, Scotland, 2009. 115

[Shahid 2009c] Z. Shahid, M. Chaumont and W. Puech. *Fast Protection of H.264/AVC by Selective Encryption.* In Proc. SinFra 2009, Singaporean-French IPAL Symposium, Fusionopolis, pages –11, Singapore, 2009. 150

[Shahid 2009d] Z. Shahid, M. Chaumont and W. Puech. *Fast Protection of H.264/AVC by Selective Encryption of CABAC for I & P frames.* In Proc. European Signal Processing Conference, pages 2201–2205, Glasgow, Scotland, 2009. 150

[Shahid 2009e] Z. Shahid, M. Chaumont and W. Puech. *Selective and Scalable Encryption of Enhancement Layers for Dyadic Scalable H.264/AVC by Scrambling of Scan Patterns.* In Proc. IEEE International Conference on Image Processing, pages 1273 – 1276, Cairo, Egypt, 2009. 169

[Shahid 2010a] Z. Shahid, M. Chaumont and W. Puech. Fast Protection of H.264/AVC by Selective Encryption of CAVLC and CABAC for I & P Frames. accepted for publication in IEEE Transactions on Circuits and Systems for Video Technology, December 2010. 150

[Shahid 2010b] Z. Shahid, M. Chaumont and W. Puech. *Over the Real-Time Selective Encryption of AVS Video Coding Standard.* In Proc. European Signal Processing Conference, Aalborg, Denmark, 2010. 150

[Shahid 2010c] Z. Shahid, M. Chaumont and W. Puech. *Selective Encryption of C2DVLC of AVS Video Coding Standard for I & P Frames.* In Proc. IEEE International Conference on Multimedia & Expo, pages 1655–1660, Singapore, 2010. 150

[Shahid 2010d] Z. Shahid, M. Chaumont and W. Puech. *Spread Spectrum-Based Watermarking for Tardos Code-Based Fingerprinting for H.264/AVC Video.* In Proc. IEEE International Conference on Image Processing, Hong Kong, September 2010. 116

[Shimizu 1988] A. Shimizu and S. Miyaguchi. *Fast Data Encipherment Algorithm FEAL.* In Proc. International Conference on Theory and Application of Cryptographic Techniques, pages 267–278, Berlin, Heidelberg, 1988. 57

[Skoric 2006] B. Skoric, T. Vladimirova, M. Celik and J. Talstra. *Tardos Fingerprinting Is Better Than We Thought.* CoRR, vol. abs/cs/0607131, 2006. 76

[Skoric 2008] B. Skoric, S. Katzenbeisser and M. Celik. *Symmetric Tardos Fingerprinting Codes for Arbitrary Alphabet Sizes.* Designs, Codes and Cryptography, vol. 46, pages 137–166, 2008. 76

[Solanki 2004] K. Solanki, N. Jacobsen, U. Madhow, B. Manjunath and S. Chandrasekaran. *Robust Image-Adaptive Data Hiding using Erasure and Error Correction.* IEEE Transactions on Image Processing, vol. 13, no. 12, pages 1627 –1639, December 2004. 45, 47

[Somekh-Baruch 2005] A. Somekh-Baruch and N. Merhav. *On the Capacity Game of Private Fingerprinting Systems under Collusion Attacks.* IEEE Transactions on Information Theory, vol. 51, pages 884–899, 2005. 70

[Somekh-Baruch 2007] A. Somekh-Baruch and N. Merhav. *Achievable Error Exponents for the Private Fingerprinting Game.* IEEE Transactions on Information Theory, vol. 53, pages 1827–1838, 2007. 70

[Somekh-Baruch 2008] A. Somekh-Baruch and N. Merhav. *On the Capacity Game of Private Fingerprinting Systems Under Collusion Attacks.* IEEE Transactions on Information Theory, vol. 54, pages 5263–5264, 2008. 70

[Standard 2007] Cinema Standard. *Digital Cinema Initiatives, LLC. Digital Cinema System Specification v1.1.* Rapport technique, 2007. 72

[Stinson 2005] D. Stinson. Cryptography: Theory and Practice, (Discrete Mathematics and Its Applications). Chapman & Hall/CRC Press, New York, November 2005. 58

[Stone 1996] H. Stone. *Analysis of Attacks on Image Watermarks with Randomized Coefficients.* Technical Report 96-045, NEC Research Institute, Princeton, NJ, USA, 1996. 81

[Su 2000] J. Su, J. Eggers and B. Girod. *Capacity of Digital Watermarks Subjected to an Optimal Collusion Attack.* In Proc. European Signal Processing Conference, pages 1981–1984, 2000. 78

[T1.801.03-2003 2003] ANSI T1.801.03-2003. *American National Standard for Telecommunications - Digital Transport of One-Way Video Signals - Parameters for Objective Performance Assessment*, July 2003. 57

[Tang 1996] L. Tang. *Methods for Encrypting and Decrypting MPEG Video Data Efficiently.* In Proc. ACM Multimedia, volume 3, pages 219–229, New York, NY, USA, 1996. 62, 63

[Tardos 2003] G. Tardos. *Optimal Probabilistic Fingerprint Codes.* In Proc. ACM symposium on Theory of computing, pages 116–125, New York, NY, USA, 2003. 73, 74, 76, 77

[Trappe 2003] W. Trappe, M. Wu, Z. Wang and L. Liu. *Anti-Collusion Fingerprinting for Multimedia.* IEEE Transactions on Signal Processing, vol. 51, pages 1069–1087, 2003. 70, 73, 76

[Wagner 1983] N. Wagner. *Fingerprinting.* In Proc. IEEE Symposium on Security and Privacy, pages 18–22, 1983. 72

[Wang 2000] Y. Wang, S. Wenger, J. Wen and A. Katsaggelos. *Error Resilient Video Coding Techniques.* IEEE Signal Processing Magazine, vol. 17, no. 4, pages 61 –82, July 2000. 61, 140

[Wang 2004a] Q. Wang, D. Zhao, S. Ma, Y. Lu, Q. Huang and W. Ga. *Context-based 2D-VLC for Video Coding.* In Proc. IEEE International Conference on Multimedia and Expo, pages 89–92, June 2004. 27, 29, 128, 141

[Wang 2004b] Z. Wang, A. Bovik, H. Sheikh and E. Simoncelli. *Image Quality Assessment: From Error Visibility to Structural Similarity.* IEEE Transactions on Image Processing, vol. 13, pages 600–612, 2004. 33

[Wang 2006] Q. Wang, D. Zhao and W. Gao. *Context-based 2D-VLC Entropy Coder in AVS Video Coding Standard.* Journal of Computer Science and Technology, vol. 21, no. 3, pages 315–322, 2006. 28

[Wang 2010] X. Wang, N. Zheng and L. Tian. *Hash Key-Based Video Encryption Scheme for H.264/AVC.* Signal Processing: Image Communication, vol. 25, no. 6, 2010. 62, 63

[Watson 1993] A. Watson. *DCT Quantization Matrices Visually Optimized for Individual Images.* In Proc. SPIE Society of Photo-Optical Instrumentation Engineers, volume 1913, pages 202–216, September 1993. 46, 50

[Watson 1997] A. Watson, G. Yang, J. Solomon and J. Villasenor. *Visibility of Wavelet Quantization Noise.* IEEE Transactions on Image Processing, vol. 6, no. 8, pages 1164 –1175, August 1997. 46

[Wen 2002] J. Wen, M. Severa, W. Zeng, M. Luttrell and W. Jin. *A Format-Compliant Configurable Encryption Framework for Access Control of Video.* IEEE Transactions on Circuits and Systems for Video Technology, vol. 12, no. 6, pages 545–557, June 2002. 61, 62, 64

[Whitman 2007] M. Whitman and H. Mattord. Principles of information security. Course Technology Press, Boston, MA, USA, 2007. 56

[Wiegand 2003] T. Wiegand, G. Sullivan, G. Bjntegaard and A. Luthra. *Overview of the H.264/AVC Video Coding Standard.* IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, pages 560–576, 2003. 22

[Wiegand 2007] T. Wiegand, G. Sullivan, J. Richel, H. Schwartz, M. Wien and eds. *Joint Scalable Video Model (JSVM) 10.* In JVT-W202, April 2007. 30

[Wolfgang 1999] R. Wolfgang, C. Podilchuk and E. Delp. *Perceptual Watermarks for Digital Images and Video.* Proc. of the IEEE, vol. 87, no. 7, pages 1108–1126, July 1999. 46

[Wu 2000] M. Wu, R. Joyce and S. Kung. *Dynamic Resource Allocation via Video Content and Short-Term Traffic Statistics.* In Proc. IEEE International Conference on Image Processing, volume 3, pages 58–61, 2000. 47, 61

[Wu 2001] C. Wu and C. Kuo. *Efficient Multimedia Encryption via Entropy Codec Design.* In Proc. SPIE Symposium on Electronic Imaging, pages 128–138, 2001. 140

[Wu 2003a] M. Wu and B. Liu. *Data Hiding in Images and Video: Part I - Fundamental Issues and Solutions.* IEEE Transactions on Image Processing, vol. 12, no. 6, pages 685–695, June 2003. 47

[Wu 2003b] M. Wu, H. Yu and B. Liu. *Data Hiding in Images and Video: Part II - Designs and Applications.* IEEE Transactions on Image Processing, vol. 12, no. 6, pages 685–695, June 2003. 47

[Wu 2004] M. Wu, W. Trappe, Z. Wang and K. Liu. *Collusion Resistant Fingerprinting for Multimedia.* IEEE Signal Processing Magazine - Special Issue on Digital Rights Management, pages 15–27, 2004. 70

[Wu 2005a] C. Wu and C. Kuo. *Design of Integrated Multimedia Compression and Encryption Systems.* IEEE Transactions on Multimedia, vol. 7, pages 828–839, October 2005. 37, 139, 140

[Wu 2005b] G. Wu, Y. Wang and W. Hsu. *Robust Watermark Embedding/Detection Algorithm for H.264 Video.* Journal of Electronic Imaging, vol. 14, no. 1, 2005. 48, 62, 64

[Xie 2008] F. Xie, T. Furon and C. Fontaine. *On-Off Keying Modulation and Tardos Fingerprinting.* In Proc. ACM workshop on Multimedia and security, pages 101–106, New York, NY, USA, 2008. 48, 49, 77

[Yabuta 2005] K. Yabuta, H. Kitazawa and T. Tanaka. *A New Concept of Security Camera Monitoring with Privacy Protection by Masking Moving Objects.* In Proc. Advances in Multimedia Information Processing, volume 1, pages 831–842, 2005. 62

[Yeo 1995] B. Yeo and B. Liu. *Rapid Scene Analysis on Compressed Video.* IEEE Transactions on Circuits and Systems for Video Technology, vol. 5, no. 6, pages 533 –544, December 1995. 61, 140

[Zeng 2003] W. Zeng and S. Lei. *Efficient Frequency Domain Selective Scrambling of Digital Video.* IEEE Transactions on Multimedia, vol. 5, pages 118–129, 2003. 62, 64

[Zhang 2009] L. Zhang, Q. Wang, N. Zhang, D. Zhao, X. Wu and W. Gao. *Context-based Entropy Coding in AVS Video Coding Standard.* Image Communication, vol. 24, no. 4, pages 263–276, 2009. 27, 29

[Ziauddin 2007] S. Ziauddin, I. Haq and M. Khan. *Method and System for Fast Context based Adaptive Binary Arithmetic Coding*, 2007. 123

[Ziv 1977] J. Ziv and A. Lempel. *A Universal Algorithm for Sequential Data Compression.* IEEE Transactions on Information Theory, vol. 23, pages 337–343, 1977. 11

[Zou 2009] D. Zou and J. Bloom. *H.264/AVC Substitution Watermarking: A CAVLC Example.* In Proc. SPIE, Media Forensics and Security XI, volume 7254, page 9, San Jose, CA, USA, 2009. 48, 50