AN IMAGE ENCRYPTION ALGORITHM ROBUST TO POST- ENCRYPTION BITRATE CONVERSION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

SADIK BAHAETTİN AKDAĞ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2006

Approval of the Graduate School of Natural and Applied Sciences

_____

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

_____

Prof. Dr. İsmet Erkmen
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

_____

Assist Prof. Dr. Çağatay Candan
Supervisor

Examining Committee Members

Assoc. Prof. Dr. Gözde Bozdağı Akar	(METU, EEE)	_____
Assist. Prof. Dr. Çağatay Candan	(METU, EEE)	_____
Assoc. Prof. Dr. Aydın Alatan	(METU, EEE)	_____
Assist. Prof. Dr. Afşar Saranlı	(METU, EEE)	_____
Barış Üçüncü (M.Sc.)	(ASELSAN)	_____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : Sadık Bahaettin Akdağ
Signature :

ABSTRACT

AN IMAGE ENCRYPTION ALGORITHM ROBUST TO
POST-ENCRYPTION BITRATE CONVERSION

Akdağ, Sadık Bahaettin
M.S., Department of Electrical and Electronics Engineering
Supervisor: Assist. Prof. Dr. Çağatay Candan

September 2006, 108 pages

In this study, a new method is proposed to protect JPEG still images through encryption by employing integer-to-integer transforms and frequency domain scrambling in DCT channels. Different from existing methods in the literature, the encrypted image can be further compressed, i.e. transcoded, after the encryption. The method provides selective encryption/security level with the adjustment of its parameters. The encryption method is tested with various images and compared with the methods in the literature in terms of scrambling performance, bandwidth expansion, key size and security. Furthermore this method is applied to the H.263 video sequences for the encryption of I-frames.

Keywords: Image Encryption, JPEG, H.263

ÖZ

ŞİFRELEME SONRASI KOD ÇEVİRİMİNDEN ETKİLENMEYEN
BİR RESİM ŞİFRELEME ALGORİTMASI

Akdağ, Sadık Bahaettin
Yüksek Lisans Tezi, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Yrd. Doç. Dr. Çağatay Candan

Eylül 2006, 108 sayfa

Bu çalışmada, JPEG standardında resim verisinin şifreleme yoluyla korunması için tamsayı dönüşümü ve frekans uzayında DCT kanallarında karıştırma yapan yeni bir yöntem önerilmiştir. Yöntem literatürde olan metotlardan farklı olarak, şifrelenmiş resimlerin sıkıştırılmasına olanak sağlamaktadır. Ayrıca yöntem ayarlanabilir parametreleri sayesinde seçilebilir şifreleme/gizlilik düzeyi sunmaktadır. Yöntem değişik resimlerde test edilmiş ve literatürdeki yöntemlerle karıştırma, bant genişliği genişlemesi, anahtar büyüklüğü ve güvenlik yönlerinden karşılaştırılmıştır. Ayrıca yöntem H.263 video dosyalarına, I-karelerin şifrelenmesi için uygulanmıştır.

Anahtar Kelimeler: Resim Şifreleme, JPEG, H.263

To My Parents,

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my supervisor Assist. Prof.Dr.Çağatay Candan for his guidance, advice, criticism, encouragements and insight throughout the research.

I would like to thank my whole family for their care on me from the beginning of my life and their trust on me that I could accomplish this task.

I would like to thank my company ASELSAN A.Ş. for their support on the every phase of this work.

TABLE OF CONTENTS

PLAGIARISM iii
ABSTRACT iv
ÖZ v
ACKNOWLEDGMENTS vii
TABLE OF CONTENTS viii
LIST OF TABLES xi
LIST OF FIGURES xiii
LIST OF ABBREVIATIONS xv
CHAPTER xv
1 INTRODUCTION 1
1.1 Motivation 1
1.2 Contributed Work 3
1.3 Organization 3
2 BACKGROUND ON MULTIMEDIA STANDARDS AND ENCRYPTION 4
2.1 Digital Image 4
2.2 Digital Image Compression 5
2.3 The JPEG Standard 10
2.3.1 Lossless Predictive Coding 10
2.3.2 DCT-based JPEG Coding 12
2.4 H.263 Video Standard 15
2.5 Cryptography and Cryptanalysis 18
2.6 JPEG Based Image Encryption 20
2.6.1 Sign Scrambling Algorithm 23
2.6.2 Slice Scrambling 25
2.6.3 DCT Coefficient Scrambling 27
2.6.4 Line Scrambling 29

3 PROPOSED ENCRYPTION METHOD 31
3.1 Introduction 31
3.2 DCT Channels 33
3.3 Block Scrambling 35
3.4 Integer to Integer Transformation 38
3.5 General View 43
3.6 Summary 48
4 USAGE OF PROPOSED METHOD ON H.263 49
4.1 Introduction 49
4.2 Encrypting I-Frames 51
4.3 Sign Scrambling of Intracoded Blocks of P-frames 52
4.4 Sign Scrambling of Motion Compensation 53
4.5 General View 54
4.6 Summary 55
5 EXPERIMENTS 56
5.1 Implementation on JPEG 56
5.2 JPEG Test Images 58
5.3 Experiments and Results on JPEG 59
5.3.1 Relation Between Key Size and Encryption Quality 59
5.3.2 Effect on File Size 62
5.3.3 Effect of Integer to Integer transformation 64
5.3.4 Quality Loss on Post-Encryption Compression 68
5.3.5 Vertical & Horizontal Modes 72
5.4 Implementation on H.263 74
5.5 Test Video Sequences 75
5.6 Experiments and Results on H.263 75
5.6.1 Encryption Performance 75
5.6.2 Effect of P-Frame Sign Scrambling 78
5.6.3 Effect on Bitrate 80
6 CONCLUSION AND FUTURE WORK 83
6.1 Features of the Proposed Method 84

6.2 Main Drawbacks 86
6.3 Suggested Future Work 87
REFERENCES 88
APPENDICES 92

LIST OF TABLES

Table 2-1: Classification of Encryption Schemes 21
Table 2-2: Sign Scrambling Performance 25
Table 2-3: Slice Scrambling Performance 27
Table 2-4: DCT Coefficient Scrambling Performance 28
Table 2-5: Line Scrambling Performance 30
Table 3-1: DCT Channel Correlations of Lena Image 39
Table 3-2: Recommended Parameters 47
Table 4-1: H.263 Configurations 51
Table 5-1: Parameters for Experiment 5.3.1 60
Table 5-2: Parameters for Experiment 5.3.2 63
Table 5-3: Encryption Bitrate Overhead 63
Table 5-4: Parameters for Experiment 5.3.3 64
Table 5-5: Parameters for Experiment 5.3.5 69
Table 5-6: Quality Loss on Barb Image 70
Table 5-7: Quality Loss on Bird Image 71
Table 5-8: Parameters for Experiment 5.3.5 72
Table 5-9: Parameters for Experiment 5.6.1 76
Table 5-10: Parameters for Experiment 5.6.2 78
Table 5-11: Parameters for Experiment 5.6.3 80
Table 6-1: Comparison of Encryption Methods 84
Table A-1: Results of 5.3.4 for Zelda Image 97
Table A-2: Results of 5.3.4 for Goldhill Image 98
Table A-3: Results of 5.3.4 for Peppers Image 99
Table A-4: Results of 5.3.4 for Lena Image 100
Table A-5: Results of 5.3.4 for Bridge Image 101
Table A-6: Results of 5.3.4 for Boat Image 102

Table A-7: Results of 5.3.4 for Girls Image 103
Table A-8: Results of 5.3.4 for Mandrill Image 104

LIST OF FIGURES

Figure 1-1: Insecure Transmission..... 2
Figure 2-1: General compression model..... 7
Figure 2-2: (a) Source Encoder Model (b) Source Decoder Model..... 8
Figure 2-3: Lossless Predictive Coding Model (a) Encoder (b) Decoder 11
Figure 2-4: Block diagram of DCT-based JPEG encoder 12
Figure 2-5: Zig-zag order 13
Figure 2-6: Block diagram of DCT-based JPEG decoder 14
Figure 2-7: Block diagram of H.263 Encoder..... 16
Figure 2-8: Block diagram of H.263 Decoder..... 17
Figure 2-9: Grayscale lena image (a) original (b) encrypted..... 23
Figure 2-10: Grayscale lena image (a) encrypted (b) recovered..... 24
Figure 2-11: Grayscale lena image (a) original (b), (c), (d) encrypted..... 26
Figure 2-12: Grayscale lena image (a) original (b) encrypted..... 28
Figure 2-13: Grayscale lena image (a) original (b) encrypted..... 29
Figure 3-1: Encryption attempts to JPEG encoder..... 31
Figure 3-2: Relation between DCT Blocks and DCT Channels..... 33
Figure 3-3: 3x3 DCT channels of Lena image. 34
Figure 3-4: Block Diagram of Scrambling Algorithm 36
Figure 3-5: Scrambling method on a DCT Channel. 37
Figure 3-6: DC channels of Lena image (a) original (b), (c), (d) scrambled . 38
Figure 3-7: Block diagram of integer to integer transformation 40
Figure 3-8: Block diagram of inverse integer to integer transformation..... 42
Figure 3-9: DC channel of Lena image (a) original (b) after transformation . 43
Figure 3-10: Block Diagram of proposed Method (Encryption) 44
Figure 3-11: Encryption Block..... 45
Figure 3-12: Block Diagram of proposed Method (Decryption) 46

Figure 3-13: Decryption Block 46
Figure 4-1: Block Diagram of H.263 encoder. 50
Figure 4-2: Usage of proposed method on I-frames 52
Figure 4-3: General Block Diagram of H.263 Bitstream Encryption 54
Figure 4-4: General Block Diagram of H.263 Bitstream Decryption 55
Figure 5-1: MATLAB GUI 57
Figure 5-2: Selected test images 58
Figure 5-3: Barb image encrypted with different block sizes. 60
Figure 5-4: Bird image encrypted with different block sizes 61
Figure 5-5: Encrypted cascade on both directions. 62
Figure 5-6: Barb image encrypted with different transform coefficients 66
Figure 5-7: Bird image encrypted with different transform coefficients 67
Figure 5-8: Quality loss test configuration 69
Figure 5-9: PSNR vs. QF for barb image 70
Figure 5-10: PSNR vs. QF for bird image 71
Figure 5-11: Barb image encrypted in different directions 73
Figure 5-12: Zelda image encrypted in different directions 73
Figure 5-13: Zelda image encrypted in vertical mod with block size of 62. .. 74
Figure 5-14: Original and encrypted frames of Foreman. 77
Figure 5-15: Original and encrypted frames of Foreman. 79
Figure 5-16: Bitrates of original and encrypted sequence per frame 81
Figure 5-17: Bitrate change percentage per frame of Foreman 81
Figure 6-1: Secure Transmission 85
Figure A-1: Results of 5.3.1 94
Figure A-2: Results of 5.3.2 95
Figure A-3: Results of 5.3.4 105
Figure A-4: Original and encrypted frames of Akiyo 107
Figure A-5: Original and encrypted frames of Carphone 108

LIST OF ABBREVIATIONS

PEL	: Picture Element
ITU	: International Telecommunication Union
ISO	: Organization for Standardization
JPEG	: Joint Photographic Experts Group
MPEG	: Moving Pictures Experts Group
DCT	: Discrete Cosine Transform
GOP	: Group of Pictures
IJG	: Independent JPEG group
BPP	: Bit Per Pixel
QF	: Quality Factor
AES	: Advanced Encryption Standard
DES	: Data Encryption Standard
KLT	: Karhunen-Loeve Transform

CHAPTER 1

INTRODUCTION

1.1 Motivation

The recent advances in technology, especially in computer industry and communications, allowed potentially enormous market for distributing digital multimedia content through the Internet. However increasing number of digital documents, multimedia processing tools, and the worldwide availability of Internet access has created an ideal way to uncontrollable distribution of multimedia content [15]. A major challenge now is how to protect the intellectual property of multimedia content in multimedia networks.

To control unauthorized access to the static (not real-time streaming) multimedia content can be accomplished by means of standard AES or DES cryptosystems. But when the multimedia data is not static, encrypting the entire multimedia stream using standard encryption methods is hard to accomplish. Moreover encrypted multimedia data could be compressed intentionally to fit the channel bandwidth by the author or unintentionally by another user while transmission as shown in Figure 1-1. Therefore the standard cryptosystems is not suitable for multimedia encryption since mostly they can not handle transmission effects like compression. Since standard encryption methods are not suitable for multimedia data transmission, multimedia encryption technologies are developed to provide end-to-end security when distributing digital content over networks.

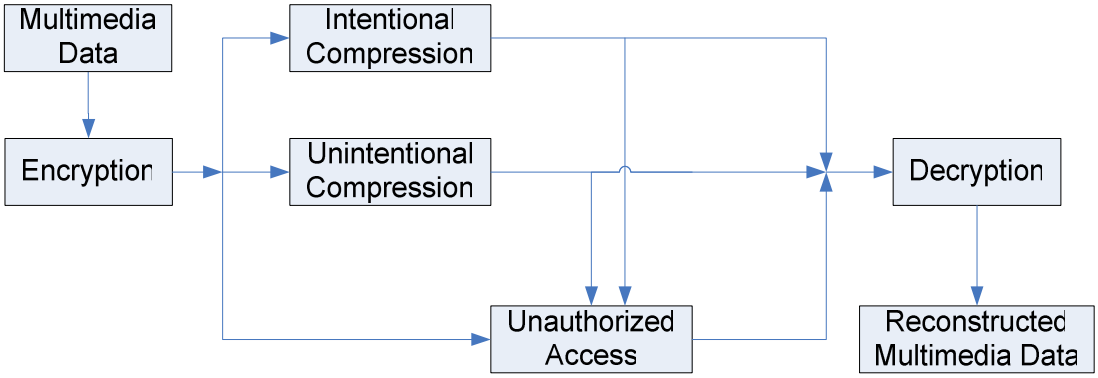


Figure 1-1: Insecure Transmission

There are number of issues to be considered while designing multimedia encryption technologies as pointed by previous works:

- Encryption (and decryption) of a multimedia data takes considerable amount of time due to large size of multimedia files. Therefore carefully selected portions of the multimedia data can be encrypted depending on their visual importance or simpler encryption methods can be used to limit computational expense [16].
- Encryption of still images can be achieved by using standard encryption methods. However application of standard techniques on raw images causes significant bandwidth expansion and these results in a major bottleneck in image communication [3].
- The security level of the multimedia encryption technologies is not required to be high like the standard text based encryption methods. As the limited lifetime and importance of streaming media considered, the security level should be set such that the encrypted media can not be compromised during its lifetime. Therefore it is obligatory to lower the security level of the encryption method to the minimum acceptable levels to achieve computational constraints such as real-time operation.
- Unlike the text data, multimedia data can be lossy compressed with very high compression ratios without losing its human visual system characteristics. The encrypted multimedia data can also be lossy

compressed intentionally to fit bandwidth limitations of the channel or unintentionally by unauthorized people. Therefore it is desired that encrypted multimedia data can be decrypted even after it has gone through lossy compression. That is encryption method should be robust to the lossy compression [7].

There is no commonly accepted multimedia encryption method to fulfill all the constraints at the optimum level. Therefore it is still an open design challenge to create a commonly accepted multimedia encryption method.

1.2 Contributed Work

This work proposes a multimedia encryption method which has partial encryption capability, very slight bit-rate overhead, acceptable security level and also which is robust to the lossy compression. This method is developed for still images and used for partially encryption of the video bit stream. Because of the enormous number of the different multimedia standards, the work should be focused on specific standards. As a still image format, today's most popular still image standard JPEG is used and as a video bitstream format H.263 used, because of the similarities of the between JPEG and H.263.

1.3 Organization

The succeeding chapters of this thesis are organized as follows: Chapter 2 gives background on multimedia standards and encryption, focusing on JPEG and H.263 standards. Chapter 3 proposes a model for the still image encryption, giving its detailed information and logical reasons of the model. Chapter 4 proposes a partial encryption method for H.263 video bit stream based on the proposed still image encryption model. Chapter 5 contains experimental results of the proposed method. The thesis is concluded in Chapter 6, with directions for future studies in this area.

CHAPTER 2

BACKGROUND ON MULTIMEDIA STANDARDS AND ENCRYPTION

2.1 Digital Image

The term image, refers to a two-dimensional light intensity function $f(x, y)$, where x and y denote spatial coordinates and the value of f at any point (x, y) is proportional to the brightness (or gray level) of the image at that point.

A digital image, which is commonly denoted by $f(n_x, n_y)$ is an image $f(x, y)$ that has been discretized both in spatial coordinates and brightness. A digital image can be considered as a matrix whose row and column indices identify a point in the image and the corresponding matrix element value identifies the gray level at that point. Each point (n_x, n_y) is called a pixel or pel [17].

The amplitude of a digital image is often quantized to 256 levels (which can be represented by eight bits). Each level is commonly denoted by an integer with 0 corresponding to the darkest level and 255 to the brightest. A digital image of 512 x 512 pixels has a spatial resolution similar to that seen in a television frame. To have a spatial resolution similar to that of an image on 35-mm film, a spatial resolution of 1024 x 1024 pixels in the digital image is needed [18].

2.2 Digital Image Compression

An enormous amount of data is produced when a 2-D light intensity function is sampled and quantized to create a digital image. In fact, the amount of data generated may be so great that it results in impractical storage, processing, and communications requirements. Therefore to use the digital image in practical applications, it is obligatory to compress the amount of data forming the digital image.

Generally, the term data compression refers to the process of reducing the amount of data required to represent a given quantity of information. A clear distinction must be made between data and information. They are not synonymous. Various amounts data may be used to represent the same amount of information. For example, if someone is telling a story, the information of interest is the story and words are the data used to relate the information. If two individuals use a different number of words to tell the same basic story, at least one of them is using words that provide no relevant information or simply restate that which is already known. It is thus said to contain data redundancy. Data redundancy is a central issue in digital image compression.

Digital image compression addresses the problem of reducing the amount of data required to represent a digital image. The underlying basis of the reduction process is the removal of redundant data. From a mathematical viewpoint, this amounts to transforming a 2-D pixel array, which is correlated and contain data redundancy, into a statistically uncorrelated data set. The transformation is applied prior to storage or transmission of the image. At some later time, the compressed image is decompressed to reconstruct the original image or estimate of it.

In digital image compression, three basic data redundancies can be identified: coding redundancy, interpixel redundancy, and psychovisual

redundancy. Data compression is achieved when one or more of these redundancies are reduced or eliminated.

Coding redundancy occur if the gray levels of an image are coded in a way that uses more code symbols than absolutely necessary to represent each gray level. In general, coding redundancy is present when the codes assigned to gray-levels have not been selected to take full advantage of the probabilities of the levels. In most images, certain gray levels are more probable than others. A natural binary coding of their gray levels assigns the same number of bits to both most and least probable values, thus failing to minimize average number of bits and resulting in coding redundancy. The average number of bits required to represent each pixel is formulated as follows:

$$L_{avg} = -\sum p_i \log_2 p_i \text{ (bits per pixel)} \quad \textbf{2-1}$$

L_{avg} : Average number of bits.

p_i : Probability of i^{th} gray level.

L : Number of gray levels.

Interpixel redundancy is directly related to the interpixel correlations within an image. Since the value of any given pixel can be reasonably predicted from the value of its neighbors, the information carried by individual pixels is relatively small. Much of the visual contribution of a single pixel to an image is redundant; it could have been guessed on the basis of its neighbors' values. In order to reduce the interpixel redundancies in an image, the 2-D pixel array normally used for human viewing and interpretation must be transformed into a more efficient format. For example, the differences between adjacent pixels can be used to represent an image.

Psychovisually redundancy is related to the human eye not responding with equal sensitivity to all visual information. Certain information simply has less relative importance than other information in normal visual processing. This information is said to be psychovisually redundant. Unlike coding and interpixel redundancy, psychovisual redundancy is associated with real or quantifiable visual information. Its elimination is possible only because the information itself is not essential for normal visual processing. Unlike the other redundancies, the elimination of psychovisual redundant data results in a loss of quantitative information. Therefore, elimination commonly referred to as quantization and results lossy data compression.

Typically the tree discussed data redundancy elimination techniques are combined to form practical image compression systems. Figure 2-1 represents generic compression system. A compression system consists of two distinct structural blocks: an encoder and a decoder. An input image $f(x,y)$ is fed into the encoder which creates a set of symbols from the input data. After transmission over the channel, the encoded representation is fed to the decoder, where a reconstructed output image $f'(x,y)$ is generated. In general $f'(x,y)$ may or may not be an exact replica of $f(x,y)$. If it is, the system is error free or information preserving; if not, some level of distortion is present in the reconstructed image.

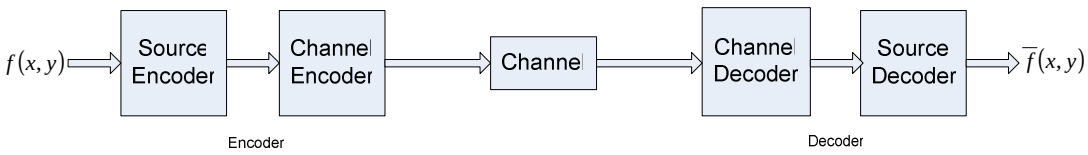


Figure 2-1: General compression model

Both the encoder and decoder shown in Figure 2-1 consist of two relatively independent functions or sub blocks. The encoder is made up of a source encoder, which remover input redundancies, and a channel encoder, which increases the noise immunity of the source encoder's output. As would be expected, the decoder includes a channel decoder followed by a source decoder. If the channel between encoder and decoder is noise free, the channel encoder and decoder are omitted, and the general encoder and decoder become the source encoder and decoder, respectively.

The source encoder is responsible for reducing or eliminating any coding, interpixel, or psychovisual redundancies in the input image. The specific application and associated fidelity requirements dictate the best encoding approach to use in any given situation. Normally, the approach can be modeled by a series of three independent operations. As Figure 2-2 shows, each operation is designed to reduce one of the three redundancies discussed.

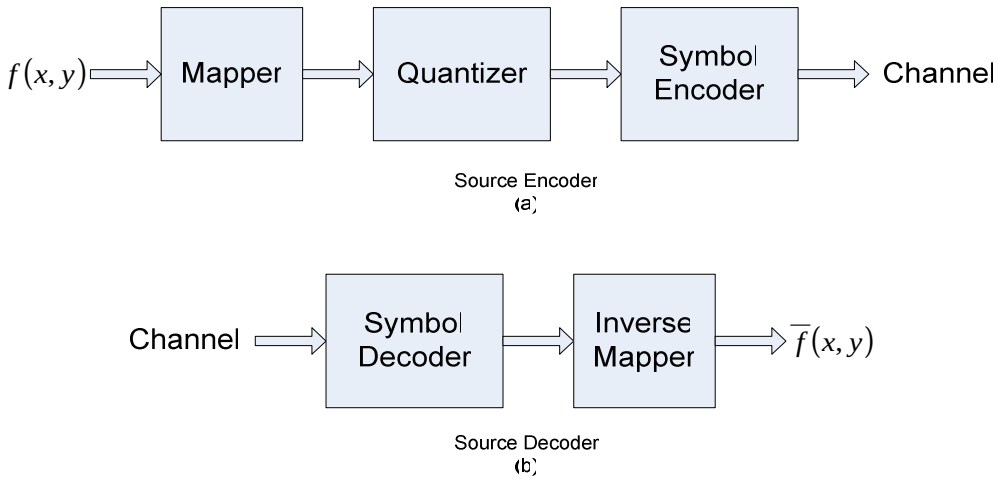


Figure 2-2: (a) Source Encoder Model (b) Source Decoder Model

In the first stage of the source encoding process, the mapper transforms the input data into a format designed to reduce interpixel redundancies in the input image. This operation generally is reversible and may or may not reduce directly the amount of data required to represent the image. Here, the mapper transforms the image into an array of coefficients, making its interpixel redundancies more accessible for compression in later stages of the encoding process.

The second stage, or quantizer block, reduces the accuracy of the mapper's output in accordance with some preestablished fidelity criterion. This stage reduces the physchovisual redundancies of the input image. This operation is irreversible. Thus it must be omitted when error-free compression is desired.
In the third and final stage of the source encoding process, the symbol coder creates a fixed- or variable length code to represent the quantizer output and maps the output in accordance with the code. The term symbol coder distinguishes this coding operation from the overall source encoding process. In most cases variable-length code is used to represent the mapped and quantized data set. It assigns the shortest code words to the most frequently occurring output values and thus reduces coding redundancy. The operation, of course, is reversible. Upon completion of symbol coding step, the input image has been processed to remove each of the three redundancies discussed.

The source decoder shown in Figure 2-2 (b) contains only two components: a symbol decoder and an inverse mapper. The blocks perform, in reverse order, the inverse operations of the source encoder's symbol encoder and mapper blocks. Because the quantization results in irreversible information loss, an inverse quantizer block is not included in the general source decoder model.

Some compression techniques normally are modeled by merging blocks that are physically separate in Figure 2-2. In the predictive compression systems the mapper and quantizer are often represented by a single block, which simultaneously performs both operations [17].

2.3 The JPEG Standard

Since the mid-1980s, members from both the International Telecommunication Union (ITU) and the International Organization for Standardization (ISO) have been working together to establish a joint international standard for the compression of multilevel still images. This effort has been known as JPEG, the Joint Photographic Experts Group. After evaluating a number of coding schemes, the JPEG members selected a DCT-based method in 1988 and it became an international standard in 1992. JPEG includes two basic compression methods: a DCT-based lossy compression method and a lossless predictive method.

2.3.1 Lossless Predictive Coding

The approach, commonly referred to as lossless predictive coding, is based on eliminating the interpixel redundancies of closely spaced pixels by extracting and coding only the new information in each pixel. The new information of a pixel is defined as the difference between the actual and predicted value of that pixel [19].

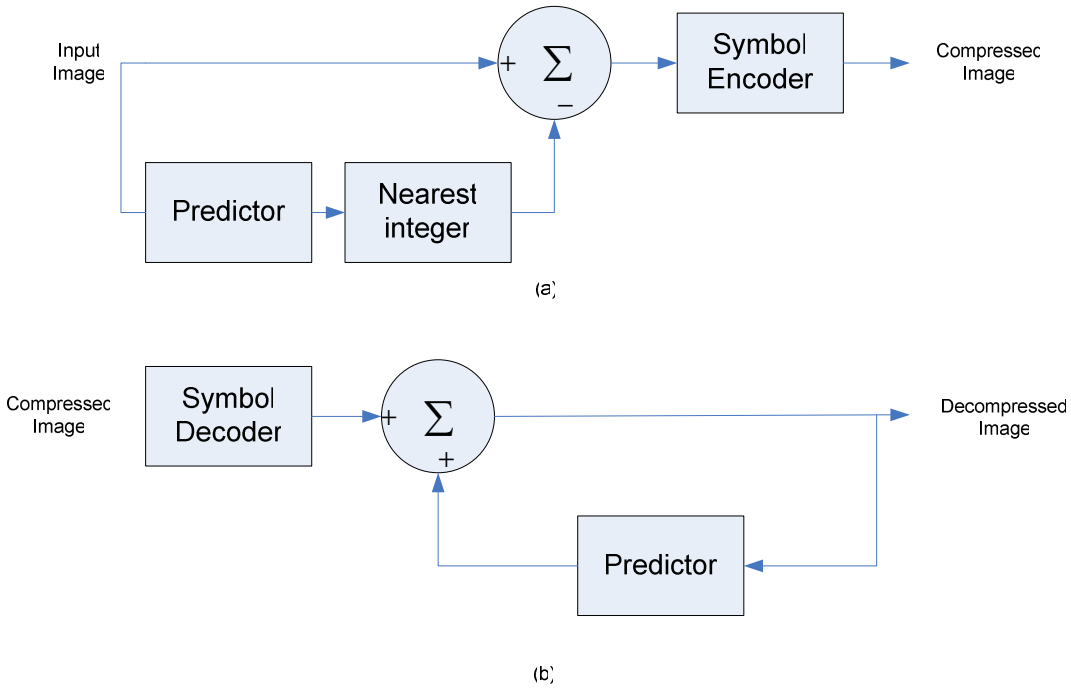


Figure 2-3: Lossless Predictive Coding Model (a) Encoder (b) Decoder

Figure 2-3 shows the basic components of a lossless predictive coding system. The system consists of an encoder and a decoder, each containing an identical predictor. As each successive pixel of the input image introduced to the encoder, the predictor generates the anticipated value of that pixel based on some number of past inputs. The output of the predictor is then rounded to the nearest integer and used to form the difference or prediction error. Prediction error is coded using a variable-length code (by the symbol encoder) to generate the next element of the compressed data stream. The decoder of Figure 2-3 reconstructs predictor error, from the received variable-length code words and performs the inverse operation.

The amount of compression achieved in lossless predictive coding is related directly to the entropy reduction that results from mapping the input image into the prediction error sequence. Because great deal of interpixel redundancy is removed by the prediction and differencing process, the probability density function of the prediction error is highly peaked at zero

and characterized by a relatively small variance in comparison to the input gray level.

2.3.2 DCT-based JPEG Coding

Figure 2-4 shows a block diagram of the DCT-based JPEG encoder for an image with a single color component (grayscale). For color images, the process is repeated for each of the color components.

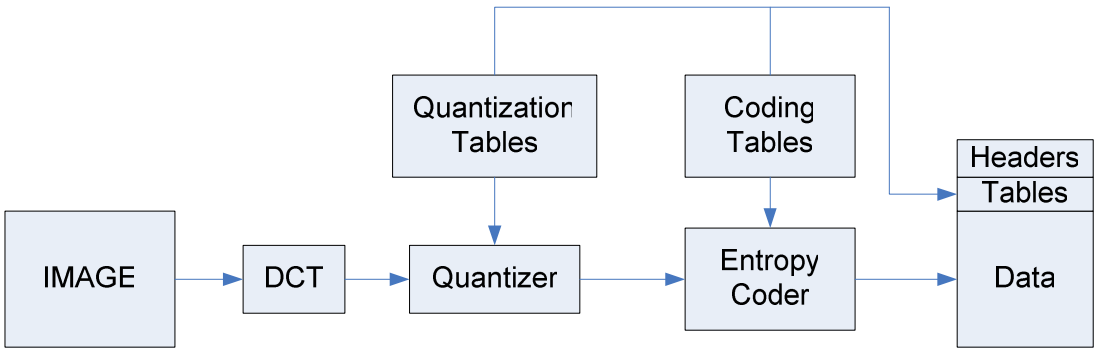


Figure 2-4: Block diagram of DCT-based JPEG encoder

From Figure 2-4, the image is first divided into nonoverlapping blocks. Each block has 8x8 pixels. If any of the dimensions of the image is not multiple of eight, then the pixels of the last row or the last column in the image are duplicated appropriately.

Each block is transformed into the frequency domain by a 2-D DCT. The standard does not specify a unique DCT algorithm. Consequently users may choose the algorithm that is best suited for their applications. The DCT output coefficients are quantized and entropy coded.

The principle source compression in DCT-based coders is the quantizer. JPEG requires an 8x8 quantization matrix for each image component to be

compressed; however, multiple components can share the same quantization matrix. Each element of the quantization matrix can be any integer value between 1 and 255 and defines the quantization step for the corresponding DCT coefficients. A quantization matrix with all ones will result in nearly lossless compression (all the loss will be due to round-off errors). In general the coefficients close to the DC (or (0, 0) coefficient) should have bigger quantization values to achieve better compression.

The first stage of the entropy coder is either a predictive coder for the DC coefficients or run-length coder for the AC coefficients. Due to the high correlation of DC values among adjacent block, JPEG uses differential coding for DC coefficient. That is the DC component is coded after differencing it from adjacent DC coefficient.

AC coefficients are processed in zig-zag order. Figure 2-5 shows the zig-zag ordering of elements in 8x8 matrix. Zig-zag ordering allows for more efficient operation of the run-length coder. A run-length coder yields the value of the next nonzero AC coefficient and a run; that is, the number of zero AC coefficients preceding this one. Hence each nonzero AC coefficient can be described by the pair (run/size, amplitude). If after a nonzero AC value all the remaining coefficients are zero, then the special symbol (0/0) denotes an end of block (EOB).

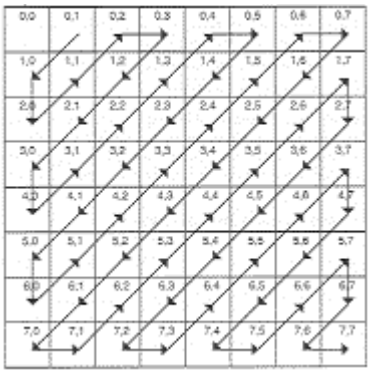


Figure 2-5: Zig-zag order

The second stage of the entropy coder is either a Huffman coder or an arithmetic coder. Since arithmetic coding is more complicated and requires license fee, Huffman coding is mostly used as an entropy coder. There is no default Huffman tables, but most applications use the Huffman tables listed in the JPEG standard.

In order to facilitate the acceptance of JPEG as an international standard and because of the various options available during JPEG encoding, the JPEG committee also defined an interchange format. This interchange format embeds image and coding parameters (type of compression, coding tables, quantization tables, image size, etc.) within the compressed bit stream. This allows JPEG compressed bit streams to be interchanged among different platforms and to be decompressed without any ambiguity.

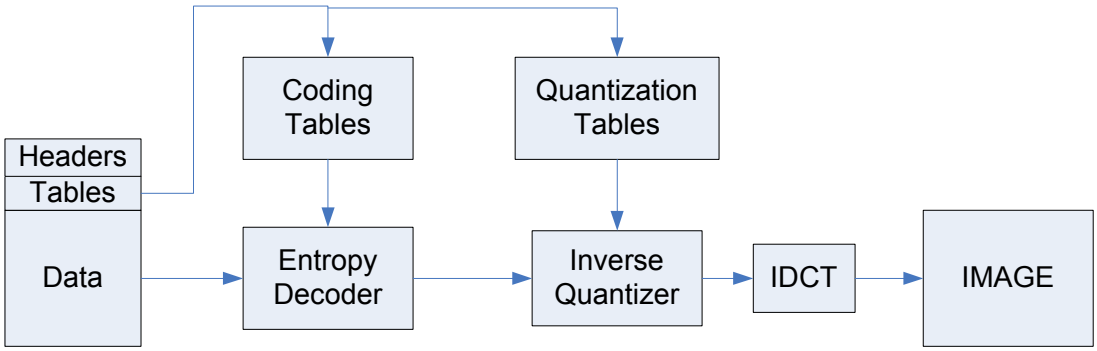


Figure 2-6: Block diagram of DCT-based JPEG decoder

Figure 2-6 shows a block diagram of a JPEG decoder. After extracting the coding and the quantization tables from the compressed bit stream, the compressed data passes through an entropy decoder. The DCT coefficients are first dequantized and then translated to the spatial domain via a 2-D inverse DCT. After a block-to-raster translation, the image is fully decoded.

In practice, images may be represented my multiple color components, each at a different resolution. For example, most color scanners generate images with red, green and blue color components (RGB). To compress a color RGB image, one approach would be applying the JPEG compression method on R, G and B components separately. Since the RGB space has significant correlation between color components, the ideal approach would be to transform the RGB image into another components representation like YUV, where each of the color components is decorrelated.

It is not necessary that all color components have the same dimensions. JPEG uses different relative sampling factors for different color components. The relative sampling factors of the color components on both vertical and horizontal directions could take integer values 1 to 4. Mostly 2 are used for relative sampling factors in both vertical and horizontal directions.

2.4 H.263 Video Standard

The H.263 standard, published by International Telecommunications Union (ITU), supports video compression (coding) for video-conferencing and video-telephony applications. H.263 is aimed particularly at video coding for low bitrates (typically 20-30kbps and above). H.263 was developed as an evolutionary improvement based on experience from H.261, the previous ITU-T standard for video compression, and the MPEG-1 and MPEG-2 standards. Its first version was completed in 1995 and it was further enhanced in projects known as H.263v2 (1998) and H.263v3 (2000).

The H.263 standard specifies the requirements for a video encoder and decoder [10]. It does not describe the encoder or decoder themselves: instead, it specifies the format and content of the encoded (compressed) stream.

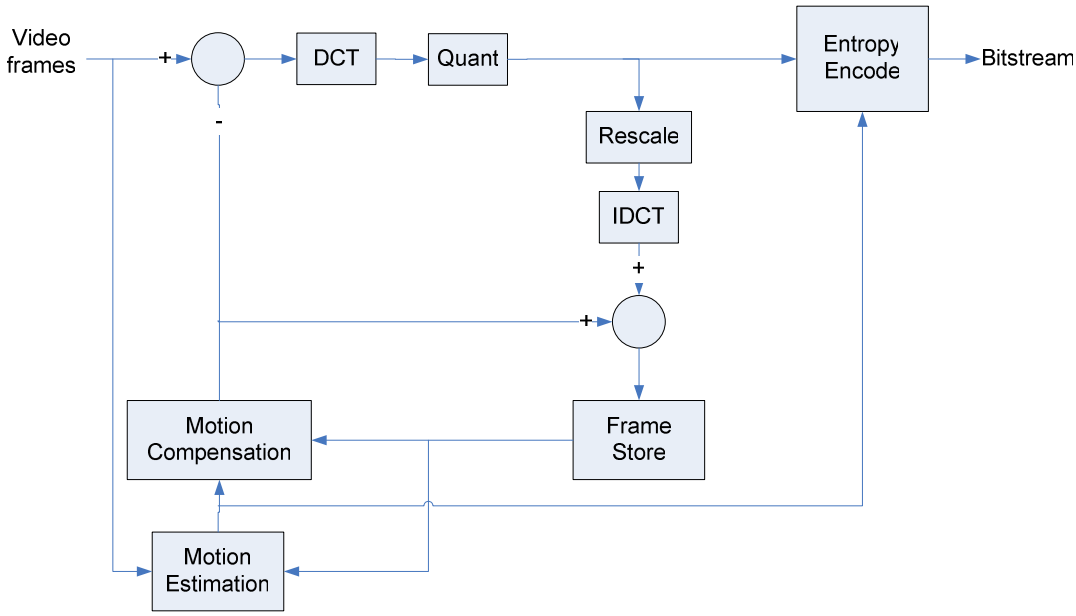


Figure 2-7: Block diagram of H.263 Encoder

Figure 2-7 shows block diagram of the H.263 encoder. The first step of this encoder is subtracting the previous transmitted frame from the current frame so that only the difference or residue needs to be encoded and transmitted. This means that areas of the frame that do not change are not encoded. Moreover it is attempted to estimate where areas of the previous frame have moved to in the current frame (motion estimation) and compensating for this movement (motion compensation). The motion estimation module compares each 16x16 pixel blocks (macroblock) in the current frame with its surrounding area in the previous frame and attempts to find a match. The matching area is moved into the current macroblock position by the motion compensator module. The motion compensated macroblock is subtracted from the current macroblock and residuals are coded.

After reducing bitrate with this blocks DCT block transforms a block of pixel values (or residual values) into a set of “spatial frequency” coefficients. That

is each block is transformed into the frequency domain by a 2-D DCT. The 2-D DCT coefficients are quantized by Quantization block which reduces the precision of each coefficient so that the near-zero coefficients are set to zero and only a few significant non-zero coefficients are left. Finally Entropy encoder block replaces frequently-occurring values with short binary codes and replaces infrequently-occurring values with longer binary codes. The result is a sequence of variable-length binary codes. These codes are combined with synchronization and control information to form encoded H.263 bitstream.

In the encoder there is also blocks called Frame store, Rescale and IDCT. Instead of simply copying the current frame into a store, the quantized coefficients are re-scaled, inverse transformed using Inverse Discrete Cosine Transform and to the motion compensated reference block to create a reconstructed frame that is placed in a store. This process is done, because only the reconstructed frame could be known by the decoder side for subtraction.

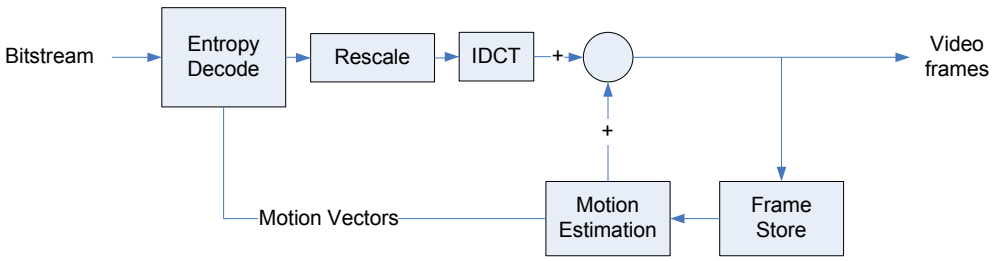


Figure 2-8: Block diagram of H.263 Decoder

Figure 2-8 shows block diagram of H.263 decoder. First block of this diagram is Entropy decoder, which takes variable-length codes and decodes in order to extract the coefficient values and motion vector information. The extracted DCT coefficients are first dequantized by rescale block and then translated to the spatial domain via a 2-D inverse DCT.

Finally motion estimation block estimates the current block with respect to the previous frame and motion vectors. The frame estimation is added to the output of the IDCT to construct the current frame.

JPEG and H.263 standards are very similar. They are both using entropy coding, quantization and DCT transformation on 8x8 blocks. In addition, MPEG2 standard is also using same blocks and transformation. Therefore it is logical to experiment H.263 or MPEG video encryption methods in JPEG first.

2.5 Cryptography and Cryptanalysis

Cryptography is the subset of science concerned in encoding data, also called encryption, so that it can only be decoded, also called decryption, by specific individuals. A system for encrypting and decrypting data is cryptosystem. Encryption usually involves an algorithm, which combines original data with key or keys to produce encrypted data, highly uncorrelated with the original data. Decryption usually involves another algorithm, which combines encrypted data with key or keys to produce original data. There are two main classes of cryptosystems: public key and secret key. Public key methods use two different keys for encryption and decryption. On the other hand, secret key encryption methods use the same key for encryption and decryption.

One of the most popular modern secret key cryptosystem is AES. The AES algorithm is essentially Rijndael [12] symmetric key cryptosystem that processes 128-bit data blocks using cipher keys with lengths of 128, 192 or 256 bits. AES copies the 128-bit data block to a two-dimensional array and using rounding, shifting, mixing operations to encrypt the data block. The security level of AES is substantial, since there are no known effective attacks discovered thus far.

Cryptanalysis is the science concerned in decrypting encrypted message in whole or in part, when the key is not known. It measures the immunity of cryptosystems to the several attacks and gives information about the security level of the systems. Depending on the amount of known information and the amount of control over the system, there are several basic types of cryptanalytic attacks:

- Ciphertext-only attack: The adversary only has access to one or more encrypted messages. The most important goal of a proposed cryptosystem is to withstand this type of attack.
- Brute force attack: This is a type of ciphertext-only attack. It is based on exhaustive key search, and for well-designed cryptosystems should be computationally infeasible.
- Known-plaintext attack: In this type of attack and adversary has some knowledge about the plaintext corresponding to the given ciphertext. This may help determine the key or a part of the key.
- Chosen-plaintext attack: Essentially, an adversary can feed the chosen plaintext into the black box that contains the encryption algorithm and the encryption key. The black box spits out the corresponding ciphertext, and the adversary may use the accumulated knowledge about the plaintext-chiphertext pairs to obtain the secret key or at least part of it.
- Chosen-ciphertext attack: Here, an adversary can feed the chosen ciphertext into the black box that contains the decryption algorithm and the decryption key. The black box produces the corresponding plaintext, and the adversary tries to obtain the secret key or a part of the key by analyzing the accumulated chiphertext-plaintext pairs.

2.6 JPEG Based Image Encryption

When dealing with still images, the security is often achieved by using the naive approach to completely encrypt the entire image. However, there are number of applications for which the naive based encryption and decryption represents a major bottleneck in communication and processing. For example, a limited bandwidth and processing power in small mobile devices calls for a different approach. Additionally, different image encryption techniques are used as a basis for selective video encryption. Therefore, apart from the standard encryption techniques, some still image techniques are proposed which requires less computational power and less bandwidth.

Most of the encryption schemes aim at JPEG, and can be naturally extended to the H.263 or MPEG because they are based on 8 x 8 block DCT followed by scalar quantization, run-length and Huffman coding. Therefore in this section, a few proposed techniques for JPEG image encryption are introduced.

Table 2-1 shows work done by Liu and Eskicioglu [20], which presents comprehensive classification that summarizes most of the presented algorithms in image and video encryption.

Table 2-1: Classification of Encryption Schemes

Type	Domain	Proposal	Encryption Algorithm	What is encrypted?
Image	Frequency Domain	Cheng & Li, 2000 [16]	No algorithm is specified	Pixel and set related significance information in the two highest pyramid levels on SPIHT
		Droogenbroeck & Benedett, 2002 [21]	DES, Triple DES and IDEA	Bits that indicate the sign and magnitude of the non-zero DCT coefficients
		Pommer & Uhl, 2003 [22]	AES	Subband decomposition structure
	Spatial domain	Cheng & Li, 2000 [16]	No Algorithm is specified	Quadtree structure
		Droogenbroeck & Benedett, 2002 [21]	Xor	Least significant bitplanes
		Prodesser, Schmidt & Uhl, 2002 [22]	AES	Most significant bitplanes
Video	Frequency domain	Meyer & Gadegast [23]	DES, RSA	Headers, parts of I-blocks, all I-blocks, I-frames of the MPEG stream
		Spanos & Maples, 1995 [24]	DES	I-frames, sequence headers and ISO end code of the MPEG stream
		Tang, 1996 [3]	Permutation, DES	DCT cefficients
		Qiao & Nahrstedt, 1997 [4]	Xor, permutation, IDEA	Every other bit of the MPEG bit stream
		Shi & Bhargava, 1998 [1]	Xor	Sign bit of DCT coefficients
		Shi, Wang & Bhargava, 1999 [2]	IDEA	Sign bit of motion vectors

Table 2-1 (Cont'd)

		Alattar, A-Regib and Al-Semari, 1999 [25]	DES	Every nth I-macroblock, headers of all the predicted macroblocks, header of very nth predicted macroblock
		Shin, Sim & Rhee, 1999 [26]	Permutation, RC4	Sign bits of DC coefficients of I pictures. Random permutation of the DCT coefficients
		Cheng & Li, 2000 [16]	No algorithm is specified	Pixel and set related significance information in the two highest pyramid leves of SPIHT in the residual error
		Wen, Severa, Zend, Luttrell & Jin, 2002 [27]	DES, AES	The information-carrying fields, either fixed length code (FLC) codewords, or variable length code (VLC) codewords
		Zeng & Lei, 2002 [7]	Permutation, xor	Selective bit scrambling, block shuffling, block rotation of the transform coefficients and JPEG motion vectors
		Wu & Mao 2002 [28]	Any modem cipher, random shuffling on bit-planes in MPEG-4 FGS	Bitstream after entropy coding, quantized values before run length coding (RLC) or RLC symbols, intra bit-plane shuffling in MPEG-4 FGS
	Spatial domain	Cheng & Li, 2000 [16]	No algorithm is specified	Quadtree structure of motion vectors and quadtree structure of residual errors
	Entropy codec	Wu & Kuo, 2000 [29]	Multiple Huffman tables, multiple state indicates in the QM coder	Encryption of data by multiple Huffman coding tables and multiple state indices in the QM coder

2.6.1 Sign Scrambling Algorithm

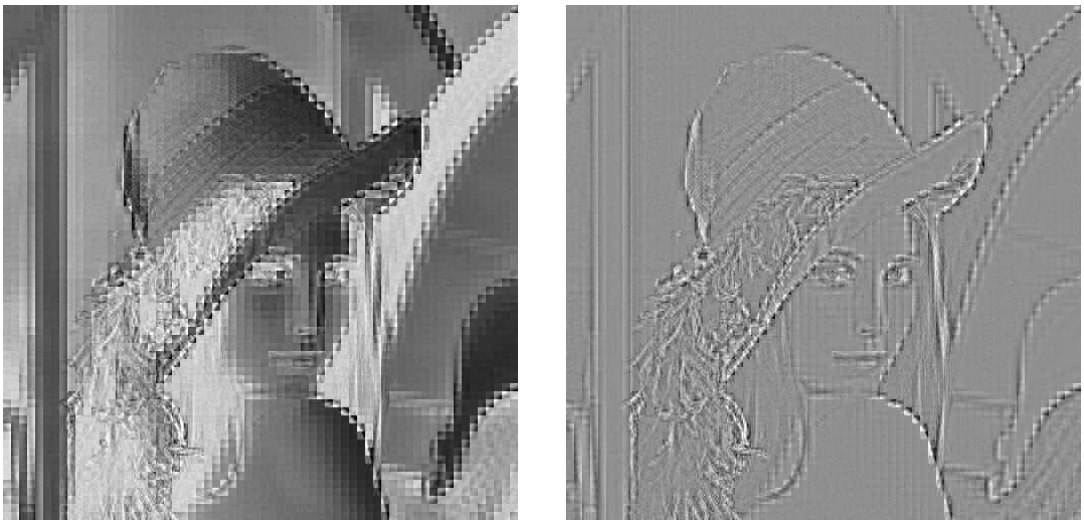
Shi and Bhargava proposed to encrypt the sign bit of every DCT coefficient in JPEG [1] and every motion vector in MPEG [2]. This approach is really fast and easy to implement. It is robust to the compression because the sign bit is not changing with respect to different quantization tables. However it lacks security and affects compression efficiency as a result of encrypting DCT coefficients before run-length and Huffman coding.

Since the sign bit could be viewed as the most significant bit of a coefficient, the results of the sign encryption expected to be good. The experiments in [1] and [2] show that the image/video would be totally unrecognizable after sign bits are randomly altered. To see the results of the sign encryption algorithm, it is implemented on MATLAB. Sign encryption is applied to Lena image with 8x8 logical scrambling matrix, which could only take 0 or 1 as a value. The results are not good as it is expected. Figure 2-9 (a) shows original image and Figure 2-9 (b) shows encrypted image.

(a) (b)

Figure 2-9: Grayscale lena image (a) original (b) encrypted

A very vague Lena contour is recognizable even in the sign encrypted image. Also if all the DC coefficients is set to 128, useful image contents can be seen as shown in Figure 2-10.



(a) (b)

Figure 2-10: Grayscale lena image (a) encrypted (b) recovered

The visual quality of recovered image can be improved using basic image processing techniques such as sharpening, contrast adjusting and thresholding. Therefore selectively encrypting sign bit of every DCT coefficient is not an effective way for image/video encryption. But because of simplicity it can be used with other encryption methods to higher security level. Table 2-2 shows the performance of the algorithm with respect to known criteria's.

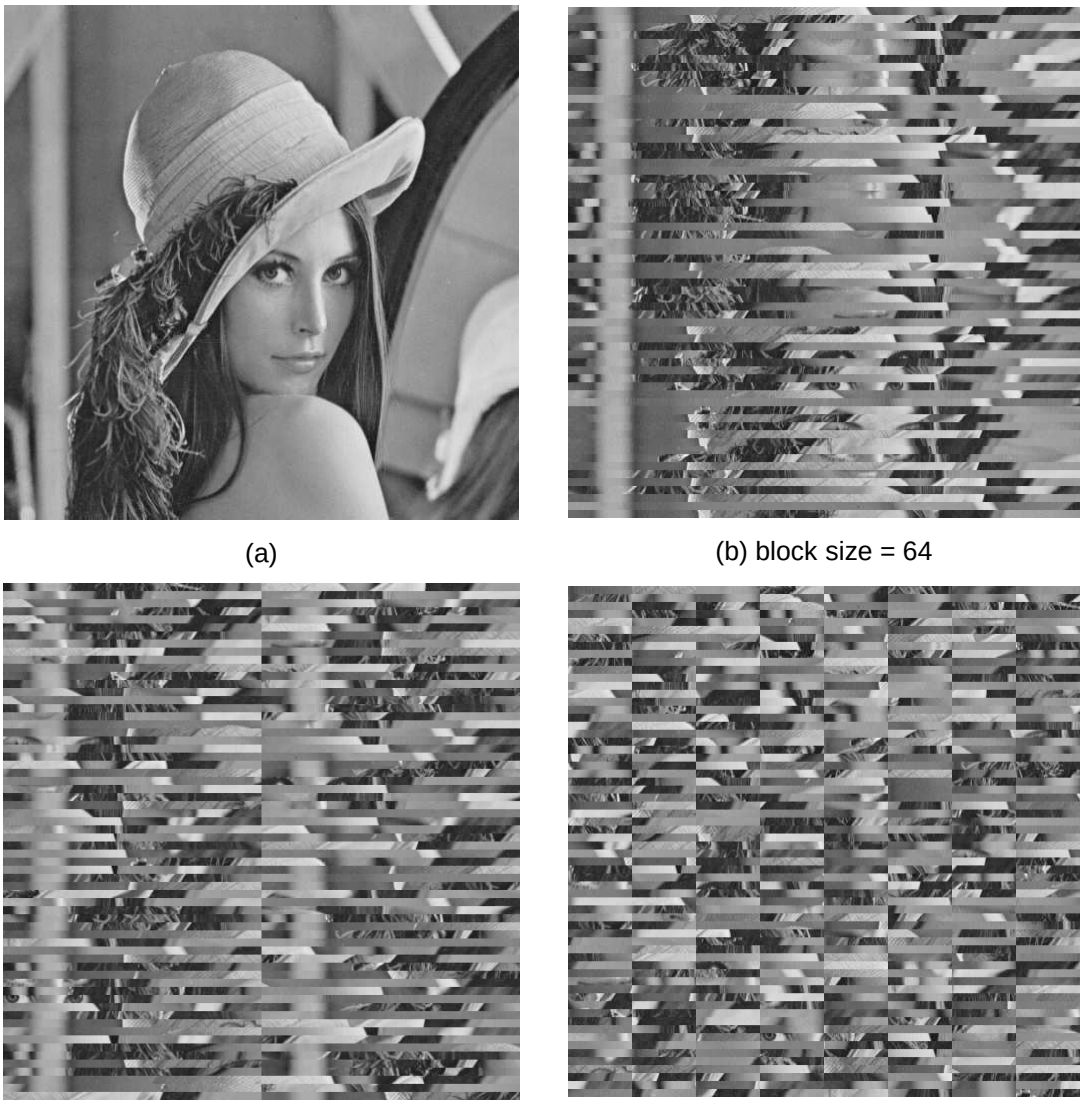
Table 2-2: Sign Scrambling Performance

Criteria	Performance
Security	Low
Key Size	Small
Overhead on Size	Small
Robustness to Bitrate Conversion	Yes
Implementation Complexity	Low
Algorithm Speed	High
Reconstruction Error	None

2.6.2 Slice Scrambling

Another approach for image encryption is slice scrambling. Wu and Kuo mentioned an encryption algorithm, which scrambles equal size of 8x8 DCT blocks with respect to a table in [7]. This approach is also easy to implement and robust to the compression, since different quantization tables will only change coefficients of the block but not the arrangement of the blocks. However it affects the compression efficiency because of placing uncorrelated DC coefficients next to each other and lacks security since one can change the block arrangement with respect to the correlation of the blocks. Moreover the block arrangement permutation should be embedded to the key, which makes key sizes very high in small block cases.

To see the results of the slice scrambling algorithm, it is implemented on MATLAB. The picture is divided into the blocks of equal sizes. And these blocks are scrambled randomly. Figure 2-11 (a) shows original image and Figure 2-11 (b), (c), (d) shows encrypted images with different block sizes.



(a) (b) block size = 64

(c) block size = 32 (d) block size = 8

Figure 2-11: Grayscale lena image (a) original (b), (c), (d) encrypted

Lena contour is recognizable in the encrypted image when block size is large. It doesn't seem very secure even visually. Moreover, the key size size gets larger with small block sizes. On the other hand, the inner blocks are not encrypted. Therefore the correlations of the originally near blocks are high. This information can be used to reconstruct original image. As a result this method also can't be used in practice. Table 2-3 shows the performance of the algorithm with respect to known criteria's.

Table 2-3: Slice Scrambling Performance

Criteria	Performance
Security	Low
Key Size	Medium - Large
Overhead on Size	Medium - Large
Robustness to Bitrate Conversion	Yes
Implementation Complexity	Low
Algorithm Speed	High
Reconstruction Error	None

2.6.3 DCT Coefficient Scrambling

Tang proposed to shuffle the DCT coefficients within an 8x8 block for JPEG/MPEG based transmission system [3]. This technique is fast and simple to implement. However it changes the statistical property (run-length characteristic) of DCT coefficients. As a result, it may increase the bitrate drastically. Moreover it lacks security.

To see the results of the DCT coefficient scrambling algorithm, it is implemented on MATLAB. The image is scrambled with 8x8 matrix, every element of which has different values from 1 to 8^2. Figure 2-12 (a) shows original image and Figure 2-12 (b) shows encrypted image.

(a) (b)

Figure 2-12: Grayscale lena image (a) original (b) encrypted

No Lena contour is recognizable in the encrypted image. Visually it is very secure. But since just inner blocks are encrypted, it has been shown in[4] that this chipper is vulnerable to frequency-based attack that exploits the property of none-zero ac coefficients that have the tendency to appear earlier in the zigzag order. In addition, it may increase the bitrate of the compressed video by as much as 50% as reported in [3] and this approach also may not be amenable to secure bitrate conversion. Therefore this method also can't be used in practice. Table 2-4 shows the performance of the algorithm with respect to known criteria's.

Table 2-4: DCT Coefficient Scrambling Performance

Criteria	Performance
Security	Medium
Key Size	Small
Overhead on Size	Large
Robustness to Bitrate Conversion	No
Implementation Complexity	Low
Algorithm Speed	High
Reconstruction Error	None

2.6.4 Line Scrambling

There are several video scrambling systems [8], [9] rely on methods of directly distorting the image in the spatial domain. These scrambling techniques are not efficient for transmitting digital video signals because they, in general, will significantly chance the statistical property of the original video signal [7].

One of the spatial domain scrambling approaches is line scrambling, which scrambles the lines of the image. To experiment spatial domain scrambling effects, line scrambling algorithm is implemented on MATLAB, just to experiment spatial domain scrambling effects. A permutation of the sequence from 1 to image height is used to rearrange the image lines. Figure 2-13 (a) shows original image and Figure 2-13 (b) shows encrypted image.

(a) (b)

Figure 2-13: Grayscale lena image (a) original (b) encrypted

No Lena contour is recognizable in the encrypted image. Visually it is very secure. But the file size of the scrambled image is 30% more than original lena image, as it mentioned in [7]. Therefore usage of this method is not proper for the limited bandwidth systems. Table 2-5 shows the performance of the algorithm with respect to known criteria's.

Table 2-5: Line Scrambling Performance

Criteria	Performance
Security	Medium
Key Size	Medium
Overhead on Size	Huge
Robustness to Bitrate Conversion	No
Implementation Complexity	Low
Algorithm Speed	High
Reconstruction Error	None

CHAPTER 3

PROPOSED ENCRYPTION METHOD

3.1 Introduction

Figure 3-1 shows the DCT-based JPEG encoder. There are three stages to apply an encryption algorithm on JPEG encoder: spatial domain, frequency domain and bitstream. Which of these stages are appropriate place to design an encryption algorithm? Which has slight bitrate overhead, robust to the bitrate conversion and acceptable security level? This is still an open question for people who are trying to design image encryption systems [13].

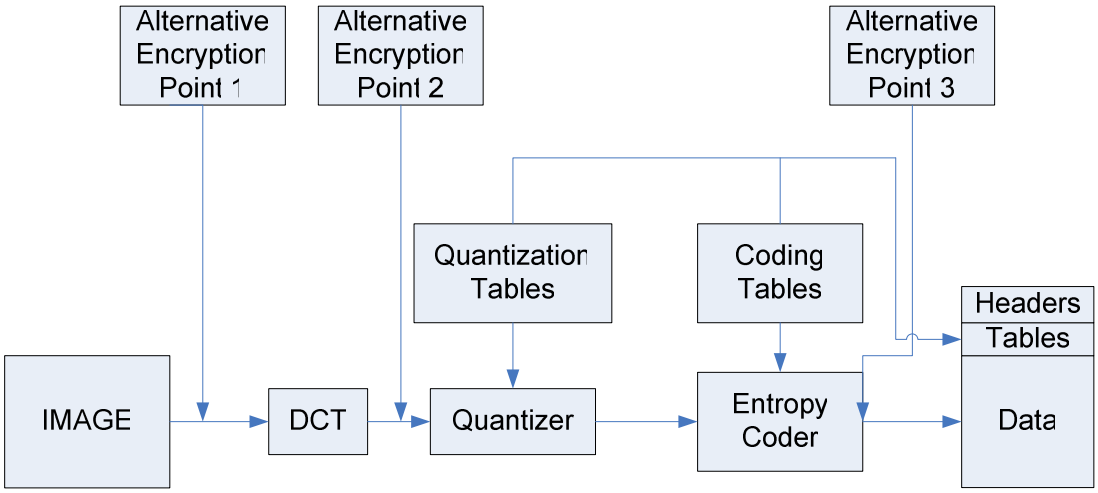


Figure 3-1: Encryption attempts to JPEG encoder

Designing an encryption algorithm at stage-1 (spatial domain) will change the image characteristics. Changing image characteristics will badly affect the working performance of JPEG encoder since the JPEG standard is designed in consideration of human visual system. That means application of an encryption algorithm on stage-1 will bring enormous bit overhead. Moreover the image in spatial domain is not an uncorrelated set of data, which means the visual information is not concentrated. Every portion of the data has equal importance for human visual system. That means digital image data can't be divided in the groups with respect to its importance. Therefore this stage is not appropriate also for selective encryption algorithms.

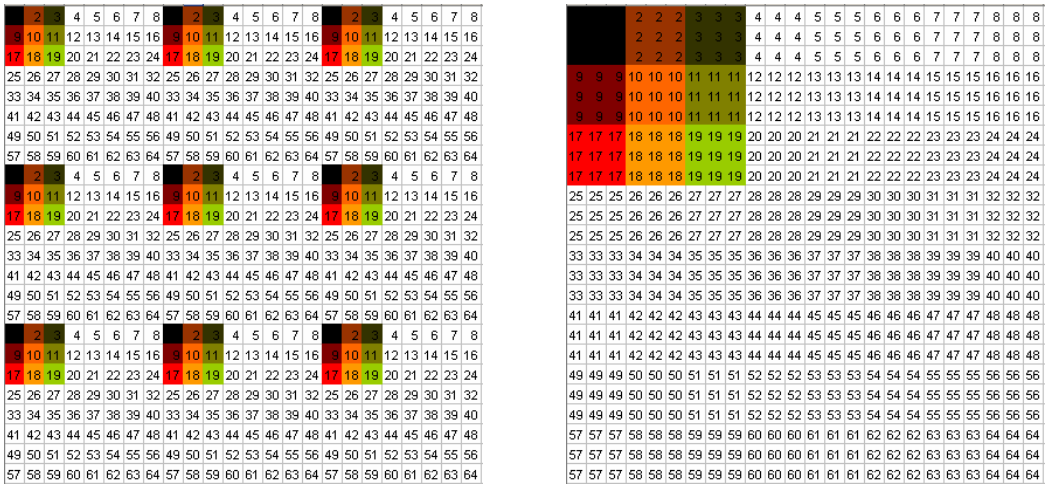
Designing an encryption algorithm at stage-2 (in the frequency domain) could be an appropriate place, if the concentrated information on the DCT coefficients near the DC coefficient is considered. But since the DCT coefficient will go through the entropy coder, which applies run-length coding to AC coefficients and differential coding to the AC coefficients, It should be hard to design an encryption algorithm which will not affected by entropy coder.

Designing an encryption algorithm at stage-3, in the entropy coder or on the bitstream, could also be an appropriate place, since the bitstream will not go through any further encoder block. But it is also hard to find a way which will be robust to the further bitrate conversion which could be applied on channel.

In the image processing point of view, stage-1 seems to be very inappropriate place and stage-3 is the mostly concern of the entropy coding design. Therefore stage-2 is used as more appropriate place for our design.

3.2 DCT Channels

DCT channel representation is collecting same frequency coefficients of the different DCT blocks together as shown on the Figure 3-2 for a hypothetical 24x24 pixel image.



(a) DCT Blocks (b) DCT Channels

Figure 3-2: Relation between DCT Blocks and DCT Channels

It is a common way to represent a JPEG image with its DCT channels like first 3x3 channels shown in Figure 3-3. A smaller version of original image can be seen on the left upper corner of the DCT channels (DC channel) and very vague contours of original image could be seen DCT channels near the DC channel.

Figure 3-3: 3x3 DCT channels of Lena image.

DCT channel representation shows that inter block DCT coefficient pattern is more important than intra block DCT coefficient pattern for human visual system. Figure 3-3 shows image characteristics are distributed in the coefficient pattern of different DCT channels. Therefore to change the image characteristics, DCT channel representation is proper place to start with.

In our design DCT channel representation is used. The reasons of using DCT channel representation are:

- Any method used in a DCT channel will change the channel coefficient pattern. And altering the channel coefficient pattern, changes the image characteristics, which makes encrypted image unrecognizable to human visual system.
- DCT channel representation gives us a selectable encryption/security capability. That means the security level of any method applied to several DCT channels could be adjustable by applying it more or less channels. This property could be used to trade-off between security level and computational performance.
- Any method applied to DCT channels most probably will not affect the run-length coding of the AC coefficients and will slightly affect the differential coding of DC coefficients. This property minimizes the bit-rate overhead of any method.

- Any compression attempt to the encrypted image could change the quantization matrix of the image. Any method used separately on DCT channels would not be affected by the compression.

Because of the reasons stated above, DCT channel representation is totally suitable for designing an secure image encryption method which is robust to bitrate conversion.

3.3 Block Scrambling

In our design a scrambling method is applied to the DCT channels separately with respect to a key. Scrambling method applied was different than the previous scrambling methods like inner-block shuffling.

Scrambling every DCT channel coefficient alone brings large computational overhead. Therefore it is easier to scramble not each coefficient but block of coefficients. Moreover it would be better to have a adjustable block size for security reasons. Mıhçak and Venkatesan proposed an algorithm for hashing based on different size of blocks [6]. This idea is used in our design as a scrambling algorithm.

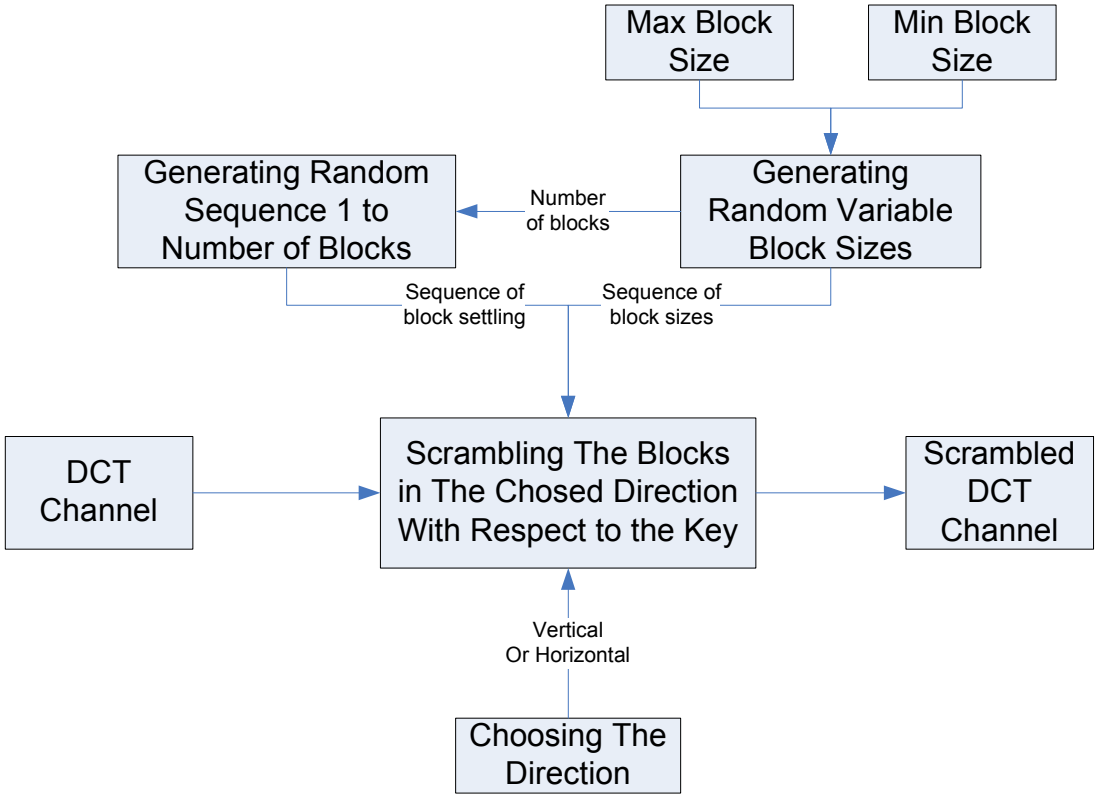


Figure 3-4: Block Diagram of Scrambling Algorithm

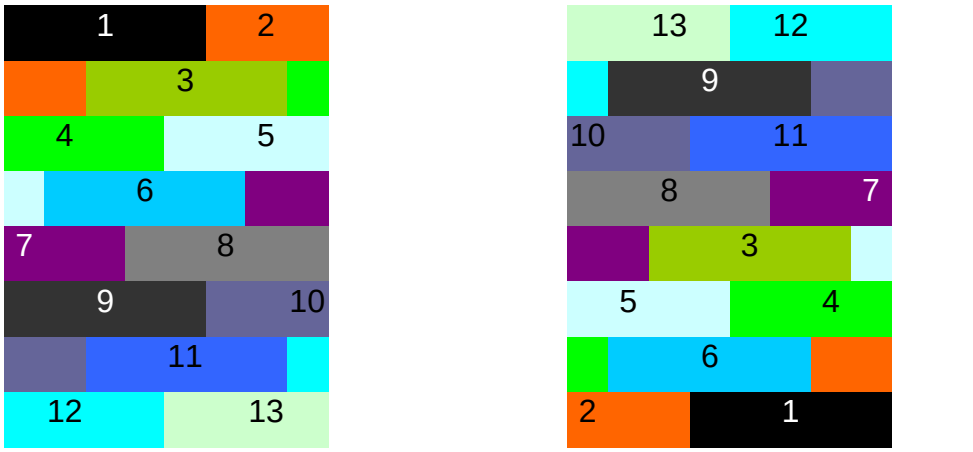
Figure 3-4 shows block diagram of scrambling algorithm of our design. Scrambling method begins with the selection of the scrambling direction, vertical or horizontal. Since one of the reasons that different images responds differently to the human visual system is the direction of the image, scrambling direction is used as a variable in our design.

After scrambling direction selection, the maximum and minimum block sizes is choused. Maximum and minimum block sizes should be in the $\left[1 \quad (\frac{width \times height}{64})\right]$ range. With the maximum and minimum block size inputs, random block sizes are generated in the $[\min \quad \max]$ range. And the number of blocks is send to another block which should generate a random permutation sequence, which is sequence of block setting, with the numbers 1 to number of blocks.

The block sizes sequence, sequence of block settling and the direction variables form the key. The DCT channel is scrambled with respect to that key in the following manner:

- Take first block size of coefficients. If the end of the column/row is reached, continue from the beginning of the next column/row
- Look the new place of the block in the block settling matrix.
- Place the block on its new place on the output matrix
- Take second block size of coefficients and continue performing pervious instruction till the end of block sizes matrix.

The scrambling algorithm used in our algorithm for a DCT channel is shown on the Figure 3-5 for a hypothetical DCT channel.



(a) A DCT Channel (b) Scrambled DCT Channel

Figure 3-5: Scrambling method on a DCT Channel.

This random pieces block scrambling method gives us various options embedded to the key, which improves the security level of the method. And this method could also be used cascade in the vertical and horizontal directions. Moreover it gives the chance of simplifying the key, i.e. by choosing maximum and minimum block sizes as image width or height.

(a) Original DC Channel

(b) Variable Block Size Scrambled

(c) Fixed Block Size in Horizontal Direction

(d) Fixed Block Size in Vertical Direction

Figure 3-6: DC channels of Lena image (a) original (b), (c), (d) scrambled

Figure 3-6 shows the DC channel of original Lena image and results of the scrambling method on that channel with different parameters. The channel is visually unrecognizable after the scrambling, but it still shares the same statistical property with the original DC channel. Even it is hard to find block sizes embedded to the key statistically; there is still information that could result in security failure. Therefore applying a scrambling method separately on DCT channels is very convenient but not sufficient way in the security manner.

3.4 Integer to Integer Transformation

Coefficients of a DCT channel are correlated. Table 3-1 shows the ratio of the correlation values $R(1)/R(0)$ in lena image DCT channels. According to the table there is high correlation in the channels near the DC channel. This correlation will result leakage on the security even after the scrambling method mentioned in 3.3. Therefore the correlation between coefficients should be decreased to increase the security. That means it is obligatory to change coefficient patterns and decrease the correlation in a DCT channel while applying channel scrambling method.

Table 3-1: DCT Channel Correlations of Lena Image

	Columns								
Rows		1	2	3	4	5	6	7	8
	1	0.779	-0.104	-0.031	-0.057	-0.070	0.010	-0.051	-0.010
	2	0.139	-0.260	0.040	-0.014	-0.021	-0.010	-0.056	-0.010
	3	-0.089	-0.158	0.132	-0.070	-0.009	0.030	0.051	0.006
	4	-0.020	-0.070	0.145	-0.075	0.062	-0.016	-0.035	0.011
	5	-0.024	-0.065	0.030	-0.092	0.092	-0.024	-0.007	-0.024
	6	-0.022	-0.041	0.006	-0.019	0.026	-0.011	-0.019	-0.028
	7	0.037	-0.028	0.018	0.005	0.060	-0.007	0.003	0.018
	8	0.078	-0.033	0.038	0.020	0.036	-0.007	0.001	-0.002

To decrease the correlation in a DCT channel, it is appropriate to use a transformation. Since JPEG DCT coefficients are defined in integer domain, the transformation should be from integer to integer.

In digital signal processing literature, it is well known that several random signals can be optimally decorrelated by the so-called Karhunen-Loeve Transform (KLT) [14]. The KLT is the unique unitary transform that diagonalizes the cross-correlation matrix of the random signals. Unfortunately, there are several problems in applying the KLT directly in practice. First and foremost, it is computationally expensive to obtain. Second, the KLT is only fixed if the statistical properties of the random signals are fixed. Therefore using KLT to decorrelate a DCT channel is practically not proper. A computationally less expensive transformation is needed to decorrelate DCT channels.

In our design an Integer to Integer transformation is applied to the variable size DCT channel blocks mentioned in 3.3. The basic idea of this transformation comes from the previous work done by Goyal in [11]. One could replace this transformation with another integer-to-integer

transformation, which also decrease the correlation on DCT channels. This part of the design is open for another suitable transformation.

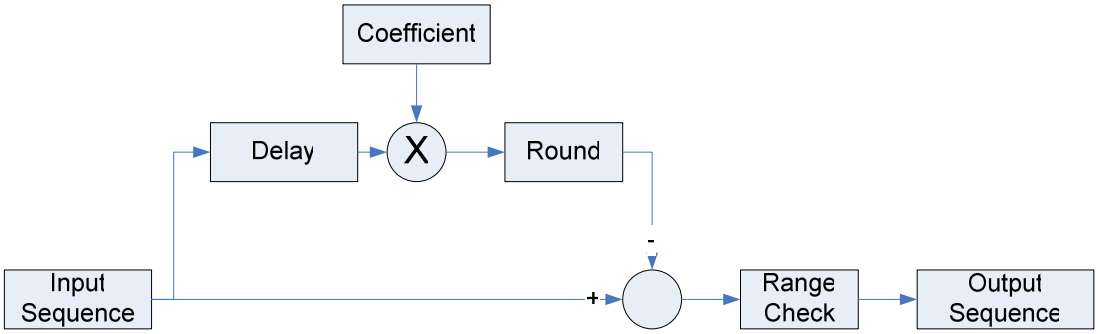


Figure 3-7: Block diagram of integer to integer transformation

Figure 3-7 shows the block diagram of the integer to integer transformation used in our design. It is a reversible transformation on the integer domain. Mathematically it can be formulated as 3-1.

$$y(1) = x(1)$$
$$y(i) = x(i) - round(c * x(i-1)) \qquad \textbf{3-1}$$

$x(n)$: the input sequence

$y(n)$: the output sequence

c : transform coefficient

This transformation is a general differencing transformation with multiplier coefficient on integer domain. Only a first order differential transformation is used in our proposed method, but it can be replaced by higher order differential transformations. This transformation is using the idea of lossless predictive JPEG coding mentioned in 2.3.1. But it is not directly subtracting

each number from the previous one. The previous number is multiplied with and coefficient in the

[-1, 1] range and rounded to the nearest integer before the subtraction. Moreover it has a range check to verify that output sequence is in the required coefficient range. If one of the coefficients of the output is not in the required range, coefficient is changed with an equal coefficient in the range. The range check operation can be formulated as 3-2. By applying this transformation to the variable size DCT channel blocks, the correlation on the blocks will decrease. And as a result the security level of the method will increase.

$$
\begin{aligned}
Ramge &= [LowerLimit, UpperLimit] \\
y &= x \text{ for } UpperLimit \geq x \geq LowerLimit \\
y &= x - UpperLimit + LowerLimit - 1 \text{ for } x > UpperLimit \\
y &= x + UpperLimit - LowerLimit + 1 \text{ for } x < LowerLimit
\end{aligned}
\qquad \textbf{3-2}
$$

This transformation seems to be irreversible because of the rounding operation in the formula, but it is reversible. The rounding operation is required since the transformation is defined in integer domain. Actually the inverse transformation is very similar to the transformation. Figure 3-8 shows the block diagram of the inverse transformation and 3-3 is its mathematical formulation. The range check block in Figure 3-8 is same as range check block in Figure 3-7.

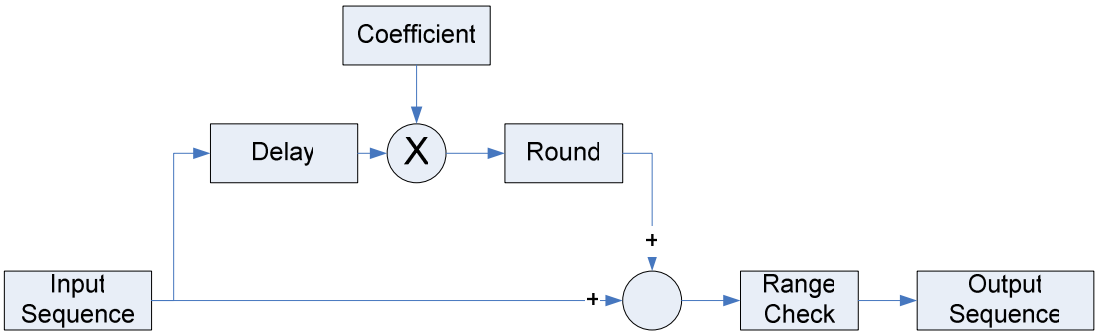


Figure 3-8: Block diagram of inverse integer to integer transformation

$$x(1) = y(1)$$
$$x(i) = y(i) + round(c * y(i-1)) \qquad \textbf{3-3}$$

$y(n)$: the input sequence

$x(n)$: the output sequence

c: transform coefficient

This transformation is effected by the bitrate conversion operation. For example, take two coefficients (x_1, x_2). In the first case, they are quantized with q. In the second case, the transformation is applied with coefficient c, then they are quantized with q, and finally reverse transformation applied. The resulting x_2 coefficients will be different. This difference is formulated as 3-4, which is not zero for arbitrary q values. The importance of this difference will be experimented in 5.3.4.

$$round((x_2 - round(x_1 * c))/q) + round(round(x_1/q) * c) - round(x_2/q) \qquad \textbf{3-4}$$

Figure 3-9 shows the DC channel of original lena image and the result of the channel after transformation. The correlation of the channel is decreased, but

that there is still some visual properties left. This is because of the usage of the just one coefficient for the whole channel. Since the transformation will be used with the channel scrambling method on 3.3, the decreased correlation will be enough for secure encryption.

(a) original (correlation = 0.77)

(b) transformed (correlation = 0.48)

Figure 3-9: DC channel of Lena image (a) original (b) after transformation

3.5 General View

In the previous sections DCT channel representation, our block scrambling strategy and our Integer to Integer transformation is explained. This three concepts form our proposed method. In this section the general block diagram of our proposed encryption method will be shown.

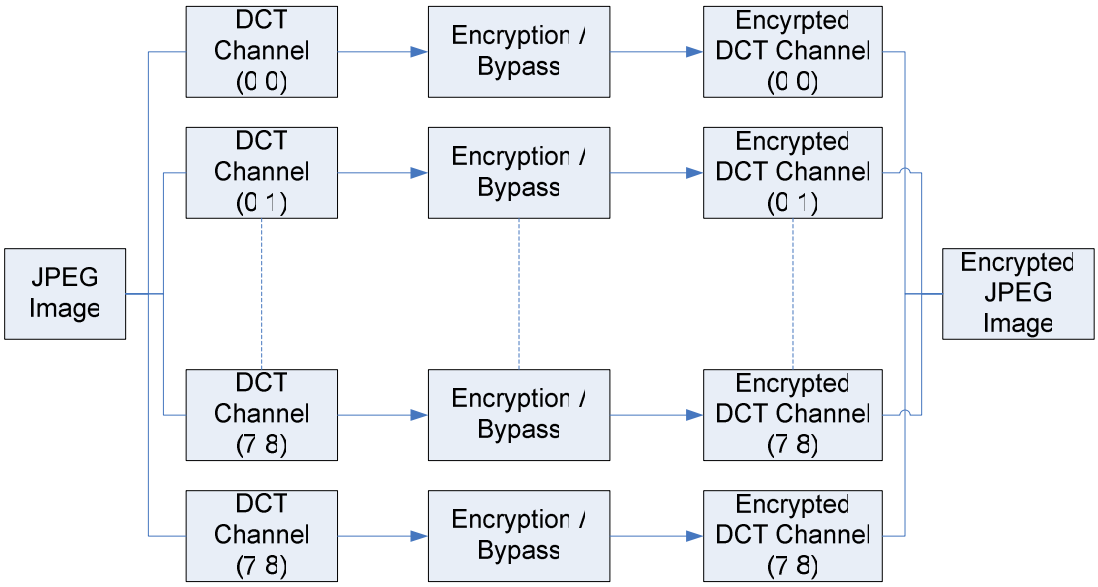


Figure 3-10: Block Diagram of proposed Method (Encryption)

Figure 3-10 shows general block diagram of our proposed encryption method. At first the JPEG image is divided into its DCT channels. The channels will pass through the encryption / bypass block separately. This encryption / bypass block will encrypt selected channels and bypasses other channels. Since the JPEG image energy on DCT domain is concentrated on the coefficients near to the (0, 0) DCT channel, encrypting channels near to the (0, 0) channel is more important than encrypting channels near the (8, 8) channel. Therefore our method inputs which channels to encrypt starting from the (0, 0) channel.

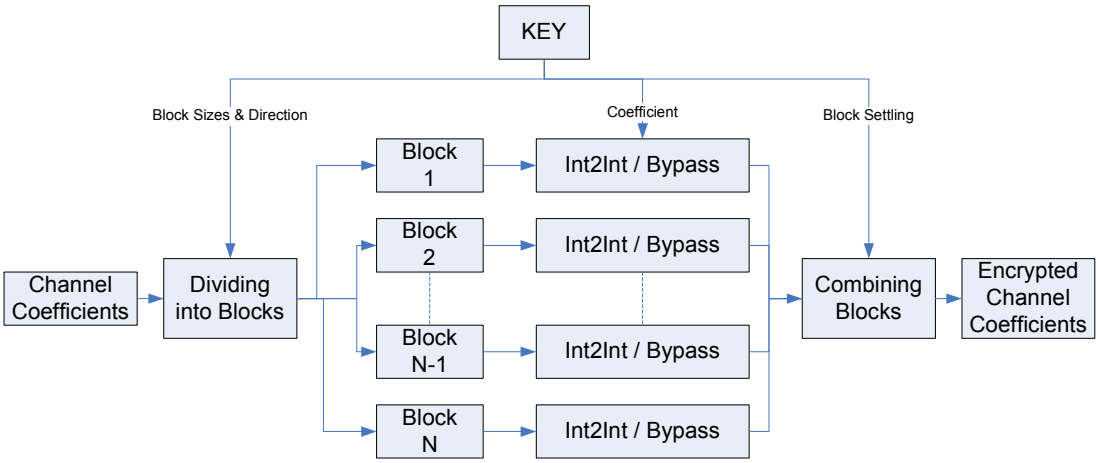


Figure 3-11: Encryption Block

Figure 3-11 shows the encryption block in Figure 3-10. In the encryption block, at first the channel coefficients are separated into blocks with respect to the given block sizes and direction. Given block sizes could be random or fixed. Then the blocks will go through the integer to integer transformation explained in 3.4 or the transformation is bypassed. This is done with respect to the key. Finally the transformed blocks are combined with respect to the block settling matrix and constitute the encrypted channel coefficients.

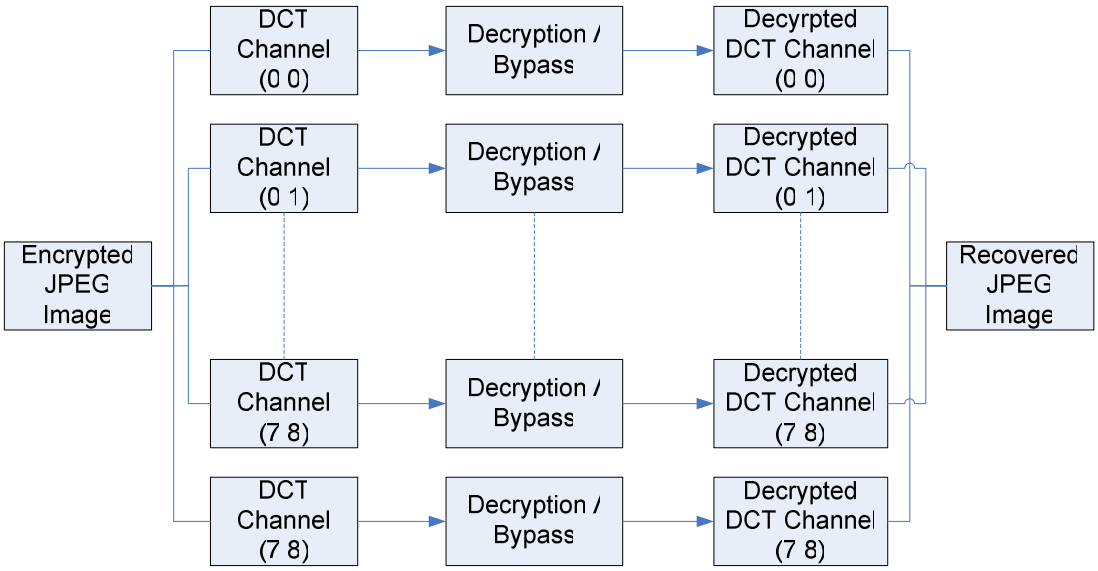


Figure 3-12: Block Diagram of proposed Method (Decryption)

Figure 3-12 shows general block diagram of our decryption method. At first the Encrypted JPEG image is divided into its DCT channels. The channels will pass through the decryption / bypass block separately. This decryption / bypass block will decrypt selected channels and bypasses other channels with respect to the key.

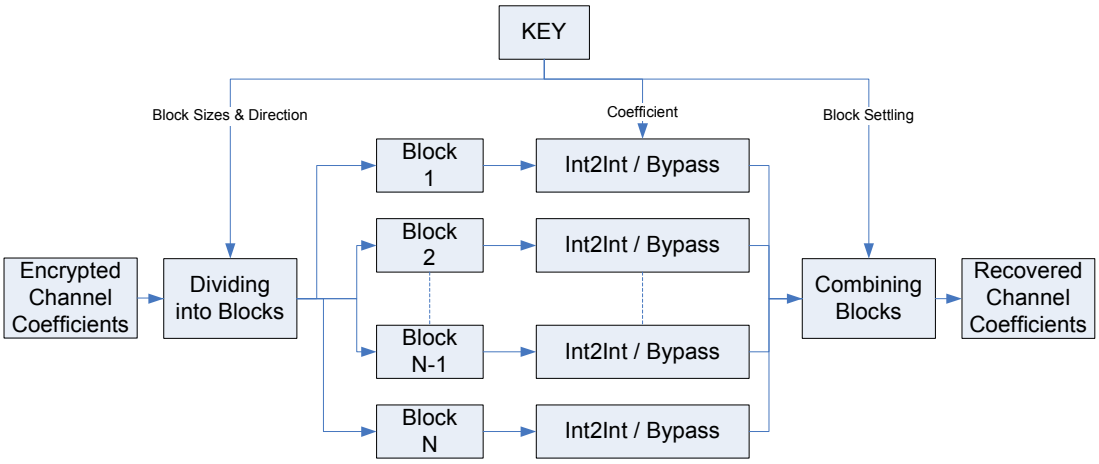


Figure 3-13: Decryption Block

Figure 3-13 shows the decryption block in Figure 3-12. In the decryption block, at first the channel coefficients are separated into blocks with respect to the given block sizes and direction in the key. Then the blocks will go through the inverse integer to integer transformation explained in 3.4 or the transformation is bypassed. This is done with respect to the key. Finally transformed blocks are combined with respect to the block settling matrix in the key and constitute the recovered channel coefficients. Since both our block scrambling strategy and Integer to Integer transformation is fully reversible, the recovered image is identical to the original image, if the encrypted image isn't compressed or changed by the channel.

The algorithm has many configuration parameters, which can be changed with respect to the different design constraints. The recommended parameters are shown on Table 3-2.

Table 3-2: Recommended Parameters

Channels to apply transformation	Only 1. channel
Channels to apply scrambling	9(3x3) or 16 (4x4) channels
Block Size	Fixed block size
Transformation coefficient	Decorrelation coefficient of channel
Direction	Cascade usage on both directions.

In practice JPEG still images are represented by three color components. But every color component goes through the same JPEG compression process. Therefore in this chapter our proposed method is explained for JPEG still images with single color components. For images with more than one color components, proposed method should be applied to every color components.

3.6 Summary

In this chapter a new encryption algorithm is proposed. This algorithm consists of DCT channel representation, variable size block shuffling, and integer to integer transformation concepts which are explained in 3.2-3.4. And at the 3.5 section general view of the proposed algorithm is presented.

CHAPTER 4

USAGE OF PROPOSED METHOD ON H.263

4.1 Introduction

In chapter 3 a new encryption algorithm for JPEG still images is proposed. This algorithm is based on DCT channel representation, variable size block shuffling, and integer to integer transformation. This algorithm can be used for still image encryption. As mentioned before, still image encryption is mostly the first step for designing video encryption systems. In this chapter our designed encryption algorithm for the video bit streams will be presented.

There are several video compression standards developed by MPEG or ITU-T such as MPEG-2, H.263 and H.261. In these standards and in most of the other video compression standards, the compression is based on 8x8 DCT to exploit the spatial redundancy like JPEG. Since our proposed encryption method is based on the channels of the 8x8 DCT channels, it could be easily and efficiently applied to the MPEG-2 or H.263 video standards. In this work H.263 standard is choused as a target video standard for our encryption algorithm, because of the license issues of the MPEG-2.

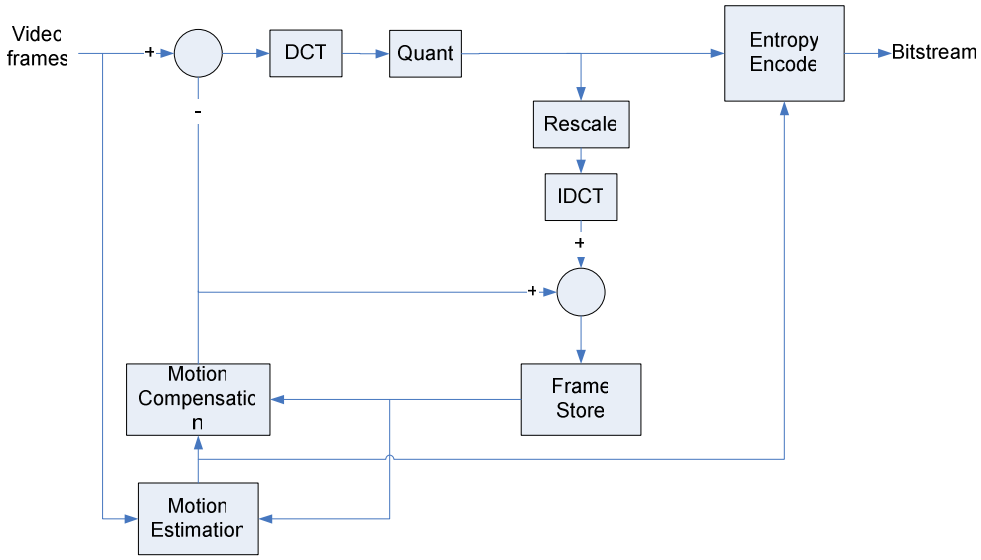


Figure 4-1: Block Diagram of H.263 encoder.

In Figure 4-1 shows block diagram of the H.263 encoder already explained in 2.4. The bitstream output of the H.263 encoder is composed of header values, motion vectors and group of pictures (GOP). Mostly the GOP of the H.263 is composed of an intracoded frame (I-frame) and some forward predictive coded frames (P-frames). The H.263 standard also has some optional frames like bidirectional predictive frame (B-frame), composition of P and B frame (PB-frame), enhanced I-frame (EI-frame) and enhanced P-frame (EP-frame). In this frame types the only fully coded frame is I-frame. All the other frames are coded with respect to I-frame, motion vectors and intracoded portions. Therefore the best way to use our approach in H.263 bitstream is by applying our proposed method to the I-frames of video sequences.

On the other hand, Zeng mentioned that encrypting the I-frames of the video bitstream is necessary but not enough for a video encryption system [7]. This means for a better video encryption performance, our proposed method should be supported by additional methods for P-frames. Therefore it is decided to apply just an easy encryption algorithm to the P-frames to support our proposed method. As a result a video encryption method is designed,

which uses our proposed method for encrypting the I-frames and sign encryption method for encrypting intracoded portions of the P-frames and motion compensation.

4.2 Encrypting I-Frames

Table 4-1 shows the different formats supported by the H.263 standard. As it can be seen from the table it is only supporting 5 predefined resolutions and using YUV color representation with relative sampling factors of 2. That means every frame of the H.263 video, including I-frame, has three different color components with different resolutions.

Table 4-1: H.263 Configurations

Picture Format	Luminance	Chroma (Blue)	Chroma (Red)	Support
SQCIF	128x96	64x48	64x48	Yes
QCIF	176x144	88x72	88x72	Yes
CIF	352x288	176x144	176x144	Optional
4CIF	704x576	352x288	352x288	Optional
16CIF	1408x1152	704x576	704x576	Optional

In Chapter 3, our proposed method is explained by using gray-scale images which has one color component. For images which has more than one color component, the proposed method should be applied to every color component of the image. But how can be the proposed method applied to the I-frames of the H.263 video, which has three color components with different resolutions?

There are two choices for applying our method to the different color components with different resolutions:

- Applying our method to every color component with different key.
- Applying our method to every color component with a single key.

First choice brings key-size overhead and more complexity. Therefore our method with a single key with respect to the chroma color components is used. And our variable shuffling block sizes are doubled for the luminance component. Figure 4-2 shows block diagram of usage of proposed method on I-frames.

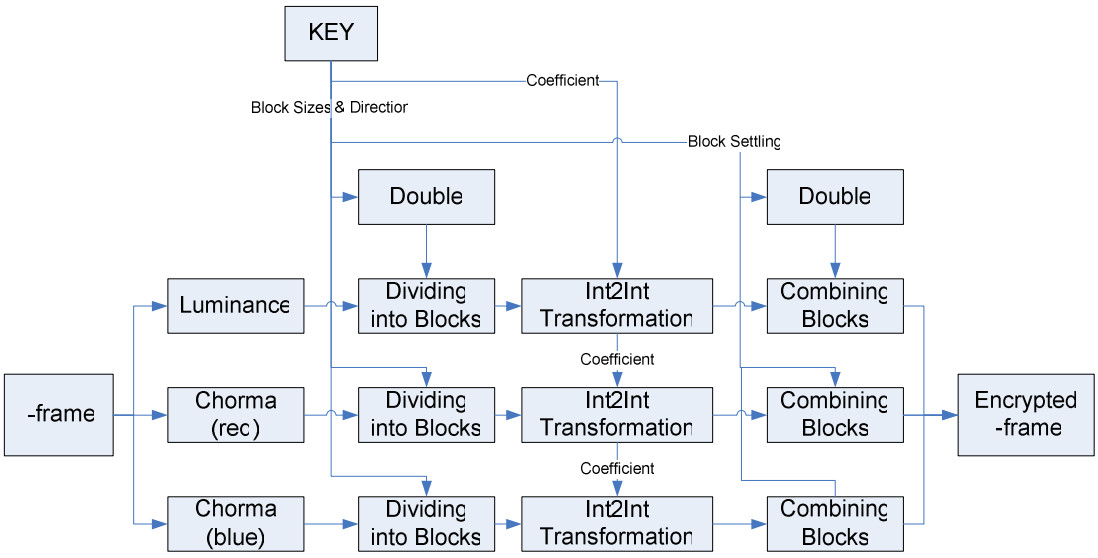


Figure 4-2: Usage of proposed method on I-frames

Apart from the different color components with different resolutions, applying our proposed method to I-frames is same as JPEG, because of the 8x8 DCT transformations of both JPEG and H.263.

4.3 Sign Scrambling of Intracoded Blocks of P-frames

Our proposed method could be easily used for I-frames of the H.263 bitstream, since every block of the frame is coded. But it can not be applied directly to the P-frames of the H.263, since P-frames are composed of motion vectors, motion compensations and intracoded blocks.

The most important part of P-frame encryption after applying our method is intracoded blocks, since intra coding erases our encrypted I-frame block and replaces it with unencrypted block. Therefore it is decided to apply sign encryption algorithm to the intracoded blocks of the P-frames.

The application of the sign encryption algorithm on the intracoded P-frame blocks is a little bit different the application on JPEG. Since the DC coefficients of the blocks do not take negative values, sign encryption could not be used for the DC coefficients. As a result a random 8x8 logical matrix, which could only take 0 or 1 as a value, is used to apply sign encryption to the AC coefficients of the intracoded blocks.

4.4 Sign Scrambling of Motion Compensation

The second important part of the P-frame encryption after applying our method is motion compensation. While motion vectors are only changing the place of the encrypted blocks of the reference frame, motion compensation is adding some additional statistical and visual characteristics to the encrypted block. That means not the motion vectors, but the motion compensation is slowly recovering the characteristics of the original block from the encrypted block. Therefore it is decided to use sign encryption method for the motion compensation.

The usage of the sign encryption on the motion compensation is same as usage on the JPEG. A random 8x8 logical matrix, which could only take 0 or 1 as a value, is used to apply and sign encryption to the coefficients of the motion compensation.

4.5 General View

In the previous sections different encryption methods used for the different parts of the H.263 bitstream is explained. In this section the general block diagram of usage of our proposed method on H.263 standard will be shown.

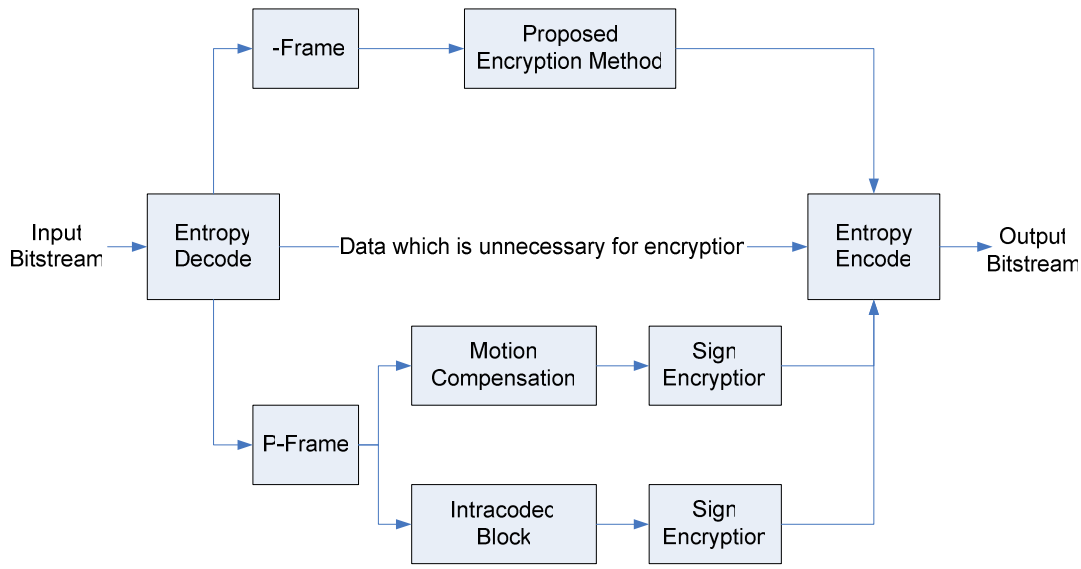


Figure 4-3: General Block Diagram of H.263 Bitstream Encryption

Figure 4-3 shows general block diagram of usage of our proposed method with sign encryption on H.263 bitstream. There are three main data paths in the figure. I-frames will input to the proposed encryption method, which outputs encrypted I-frames. The encrypted I-frames are entropy coded and written on the output bitstream. P-frames will be divided into motion compensation and intracoded block coefficients. They will input to the sign encryption blocks, which outputs encrypted motion compensation and encrypted intracoded block. The encrypted P-frame portions are entropy coded and written on the output bitstream. Every other information embedded on the input bitstream is directly written on the output bitstream.

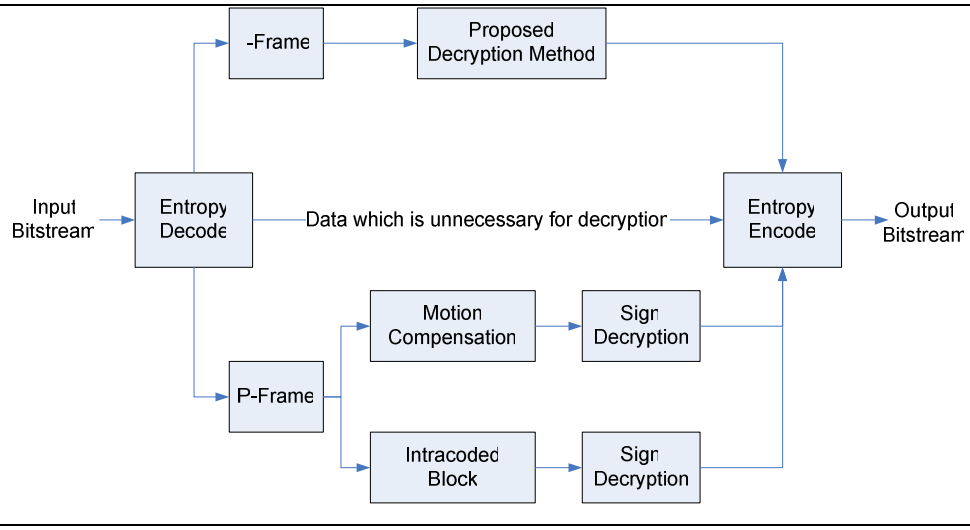


Figure 4-4: General Block Diagram of H.263 Bitstream Decryption

Figure 4-4 shows general block diagram of decryption on H.263 bitstream. The working principles are same as the encryption process. The only difference is instead of encryption blocks, there is decryption blocks.

4.6 Summary

In this chapter, how our encryption method on H.263 standard could be used is proposed. Since our method is designed for the still images, it is used on the I-frames of the H.263 bitstream as explained on 4.1. Moreover the motion compensation and intracoded blocks is encrypted by sign encryption algorithm to increase the security level and visual scrambling as explained in 4.3 and 4.4. Finally the general view of the encryption and decryption process is represented.

CHAPTER 5

EXPERIMENTS

5.1 Implementation on JPEG

The proposed method is implemented over Independent JPEG groups (IJG) free JPEG library and MATLAB. The free IJG JPEG library is used to read and write the JPEG bitstream parameters and MATLAB is used for the implementation of encryption/decryption methods and GUI.

At first DCT coefficients, quantization tables, huffman tables and header parameters of JPEG bitstream is extracted by IJG JPEG library. These parameters inputs to the encryption or decryption functions which is implemented on MATLAB. The output of the encryption or decryption functions is written to the JPEG bitstream by IJG JPEG library.

Also a graphical user interface on MATLAB is implemented, by which you can select any of the explained encryption methods and apply it to a JPEG image with different parameters Figure 5-1 shows our GUI used on the tests.

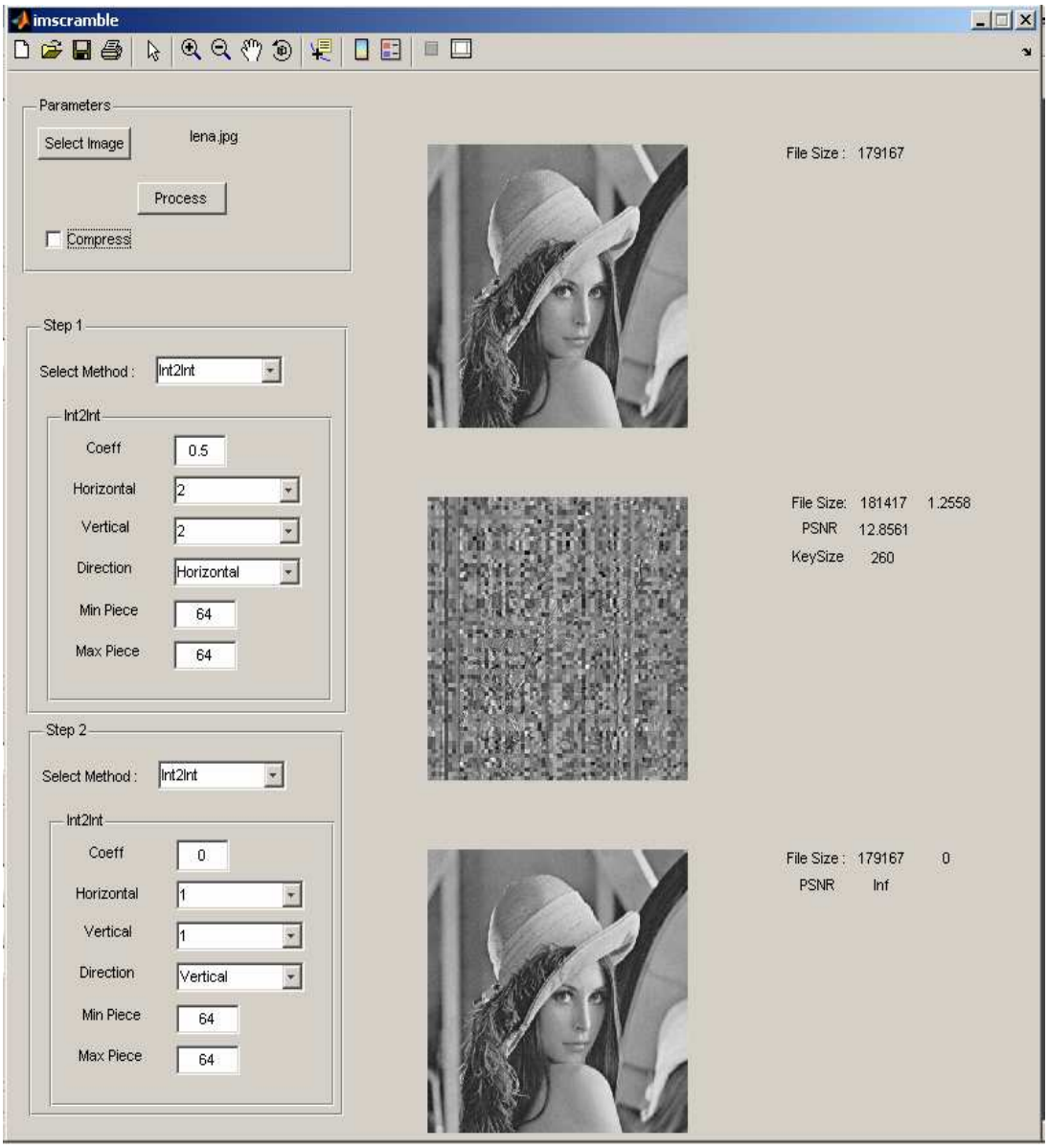


Figure 5-1: MATLAB GUI

5.2 JPEG Test Images

10 JPEG test images that were commonly used in the literature are selected for the experiments. The sequences are obtained in (or later converted to) gray-scale DCT JPEG image. They have different resolutions and different characteristics. Figure 5-2 shows the original test images.

Barb (512x512)

Zelda (512x512)

Goldhill (512x512)

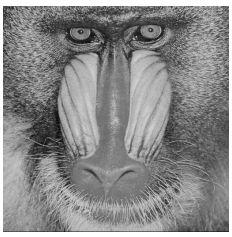
Peppers (512x512)

Bird (256x256)

Bridge (256x256)

Boat (512x512)

Girls (512x512)



Mandrill(512x512)

Lena (512x512)

Figure 5-2: Selected test images

5.3 Experiments and Results on JPEG

The transform coefficient should be selected with respect to the correlation between DCT channel coefficients to encrypt the image more efficiently. Since the proposed algorithm has no part to estimate the optimum transform coefficient for computational complexity reasons, an arbitrary integer to integer transform coefficient like 0.5 is used in most of our experiments. But the user should be careful about the coefficient, because inharmonious transform coefficient will result larger file size overhead.

In literature commonly used performance criteria's of digital image encryption algorithms are bandwidth expansion, security, key size and bitrate conversion. Therefore proposed method is tested with respect to this criteria's in the following experiments. Moreover an additional test is made to show the effect of the integer-to-integer transformation.

5.3.1 Relation Between Key Size and Encryption Quality

The proposed method has variable key size. The key size depends on the scrambling block sizes matrix and block settling matrix. Block sizes matrix depends on whether variable or fixed block sizes are used. In the fixed block size case, the block size matrix will have just a member. But in the variable block size case, block size matrix will have all the block sizes as a member. Therefore variable block size choice increases the key size. Fixed block size is used in our experiments to not increase the key size.

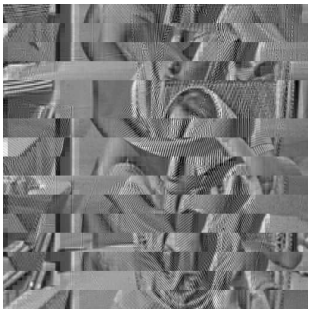
On the other hand block settling matrix will have new settling of blocks. Therefore its size depends on the $BlockSize/\mathrm{Im}\,ageSize$ ratio. That means small block sizes increases block settling matrix size. On the contrary having small block sizes will increase the security. In this experiment the key size,

which depends on block size, vs. visual encryption performance, is examined.

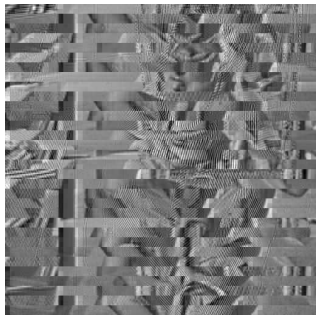
Proposed method is applied to the test images with fixed block sizes of 1/2, 1/4, 1/8, 1/16, and 1/32 times image width. The used parameters are shown on Table 5-1. The results of the barb and bird pictures are shown in Figure 5-3 and Figure 5-4.

Table 5-1: Parameters for Experiment 5.3.1

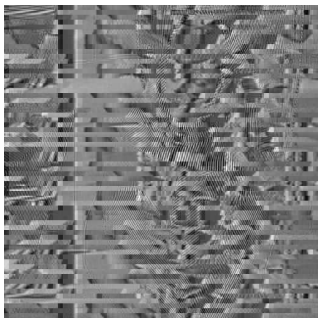
Channels to apply transformation	Only 1. channel
Channels to apply scrambling	9(3x3) channels
Block Size	Fixed block size (1/2, 1/4, 1/8, 1/16, 1/32 times image width)
Transformation coefficient	0.5
Direction	Horizontal.



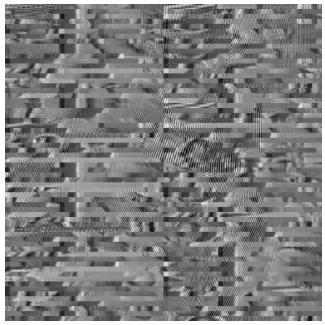
(a) Block Size: 1/2 , Key Size: 19 B



(b) Block Size: 1/4 , Key Size: 35 B

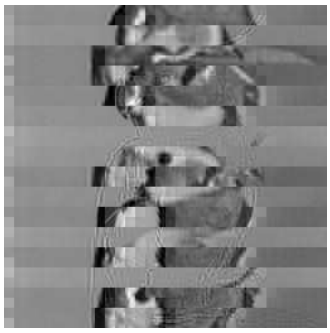


(c) Block Size: 1/8 , Key Size: 67 B

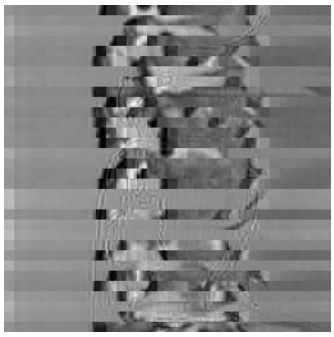


(d) Block Size: 1/16 , Key Size:131B

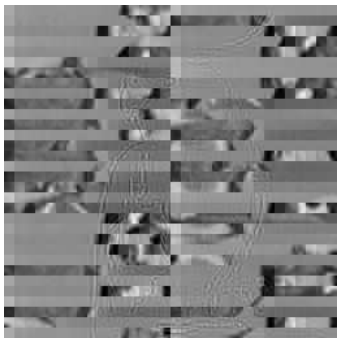
Figure 5-3: Barb image encrypted with different block sizes.



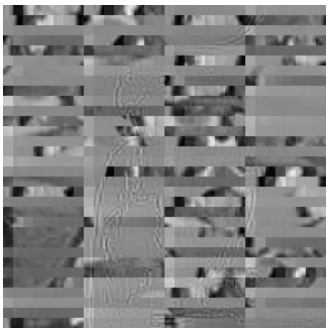
(a) Block Size: 1/4 , Key Size: 19 B



(b) Block Size: 1/8 , Key Size: 35 B



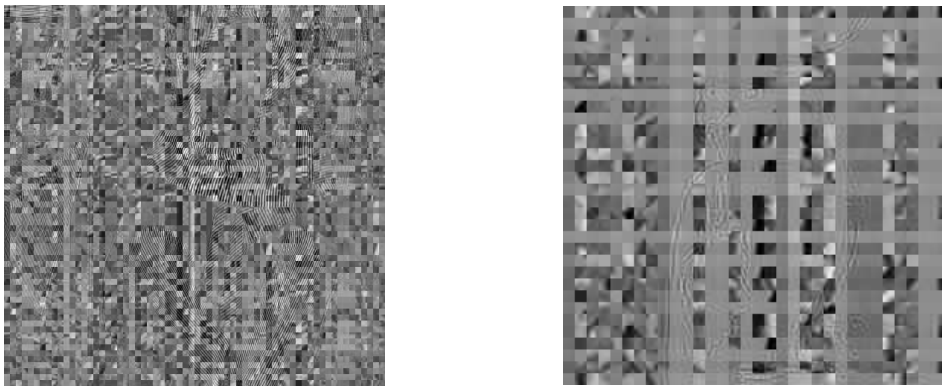
(c) Block Size: 1/16 , Key Size: 67 B



(d) Block Size: 1/32 , Key Size:131B

Figure 5-4: Bird image encrypted with different block sizes

First two cases are visually recognizable and other cases are unrecognizable. This show to achieve proper visual scrambling block size of 1/16 or 1/8 times image width should be used when the algorithm is applied on one direction. On the other hand, our method could establish more visual encryption by using it cascade on both directions with the same key. The results of the usage of our method cascade in both directions with block size of 1/8 times image width for barb image and 1/16 times image width for bird image are shown in Figure 5-5. The results of the other pictures are in appendix A.



(a) Barb (b) Bird

Figure 5-5: Encrypted cascade on both directions.

Another issue by selecting block sizes shows itself on the figures which have fixed block size values which divides the image width. Block sizes which are relatively prime with image width will result higher visual encryption and better immunity to the correlation attacks. This is since by selecting exact divides of the image width aligns blocks, which are highly correlated, in vertical direction. Even the difference can not be seen, this will make correlation attacks to the algorithm harder to success.

5.3.2 Effect on File Size

The block scrambling method and integer to integer transform used in proposed method is affecting the differential coding used for DC coefficients and run-length coding used for AC coefficients. Therefore the proposed method is changing the compression efficiency. In this experiment effect of our proposed method on file size is examined.

Proposed method is applied to the test images with parameters shown on Table 5-2 and the encrypted image file size and original image file size is recorded. The results are shown on Table 5-3.

Table 5-2: Parameters for Experiment 5.3.2

Channels to apply transformation	Only 1. channel
Channels to apply scrambling	9(3x3) channels
Block Size	Fixed block size (1/8 for 512x512 images, 1/16 for 256x256 images)
Transformation coefficient	0.5
Direction	Cascade

Table 5-3: Encryption Bitrate Overhead

Test Image	Original Image File Size (bpp)	Encrypted Image File Size (bpp)	Percentage of File Size Change
Barb	0.6772	0.6830	0.85%
Zelda	0.5721	0.5767	0.80%
Goldhill	0.7027	0.7095	0.97%
Peppers	0.6575	0.6612	0.56%
Bird	0.5268	0.5347	1.48%
Lena	0.6166	0.6201	0.55%
Bridge	0.9211	0.9298	0.94%
Boat	0.6362	0.6404	0.65%
Girls	0.5863	0.5916	0.90%
Mandrill	0.9371	0.9388	0.17%

According to the results, apart from the bird image our proposed method has less than 1% file size overhead. This is expected. Since bird image has highly correlated DCT channels, coefficient value of 0.5 is not appropriate. The result of the same experiment on bird image with coefficient 0.9 has bit overhead less than 1%. Therefore it shows that our proposed method has slight bit overhead with appropriate parameters. But this overhead will increase by choosing small scrambling block sizes or inharmonious transform coefficients.

5.3.3 Effect of Integer to Integer transformation

The integer transformation used in the design is a proper but not optimum for reducing correlation on the DCT channels. As it said before, the integer to integer transformation used could be replaced with a more appropriate transformation.

The integer to integer transformation used is very dependent on the difference coefficient parameter. The correlation in the DCT channels is suppressed by small transform coefficients on the images, those changes slightly pixel to pixel. But the other images, which have big changes pixel to pixel, require large coefficients. In this experiment the coefficient vs. correlation on channels is examined.

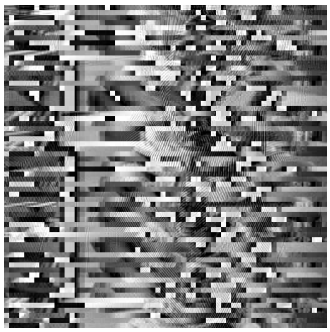
Proposed method is applied to the test images with coefficients -0.9, -0.6, -0.3, 0, 0.3, 0.5, 0.7 and 0.9. The used parameters are shown on Table 5-4. The results of the barb and camera pictures are shown in Figure 5-6 and Figure 5-7. The results of the other pictures with coefficients 0 and 0.9 are in appendix A.

Table 5-4: Parameters for Experiment 5.3.3

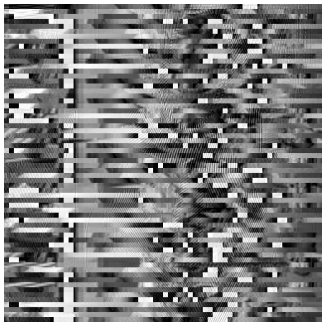
Channels to apply transformation	Only 1. channel
Channels to apply scrambling	9(3x3) channels
Block Size	Fixed block size (1/8)
Transformation coefficient	-0.9, -0.6, -0.3, 0, 0.3, 0.5, 0.7, 0.9
Direction	Horizontal.

The barb image, which is slightly changing in spatial domain, can not be recognized with most of the positive coefficients, but bird image, which has sharp changes in spatial domain, can be recognized with small positive

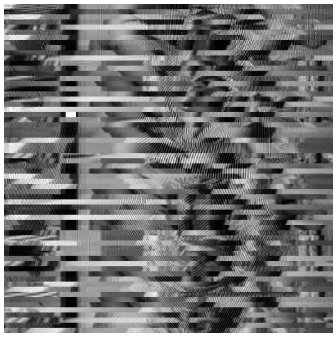
coefficients. On the other hand negative coefficients results more correlated channel, which could result in security leakage with correlation based attacks.



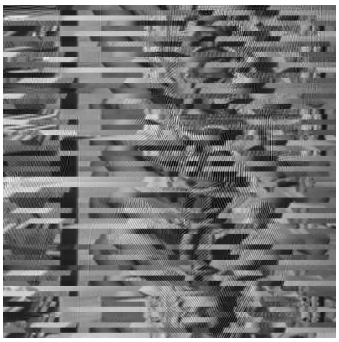
(a) -0.9



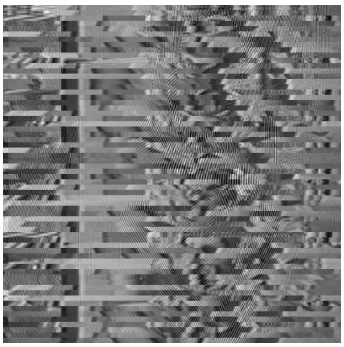
(b) -0.6



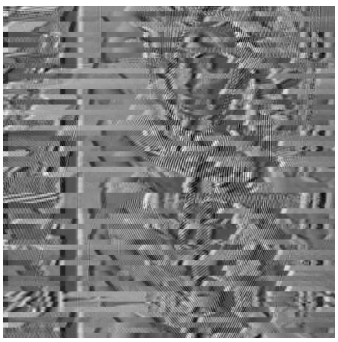
(c) -0.3



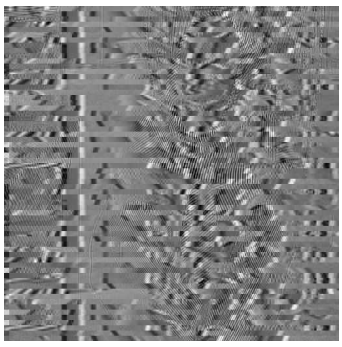
(d) 0



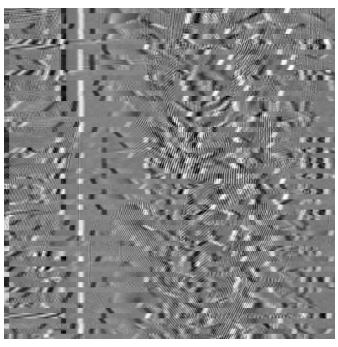
(e) 0.3



(f) 0.5



(g) 0.7



(h) 0.9

Figure 5-6: Barb image encrypted with different transform coefficients

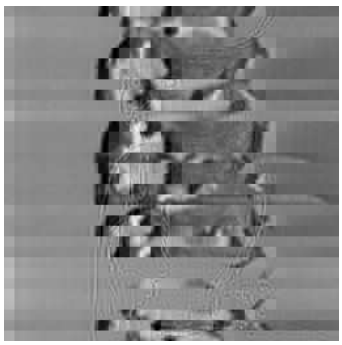
(a) -0.9

(b) -0.6

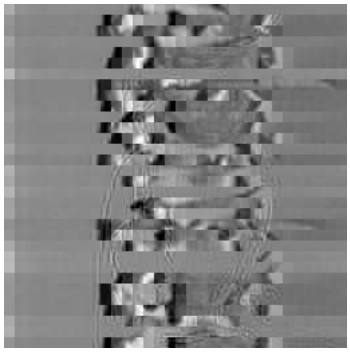
(c) -0.3

(d) 0

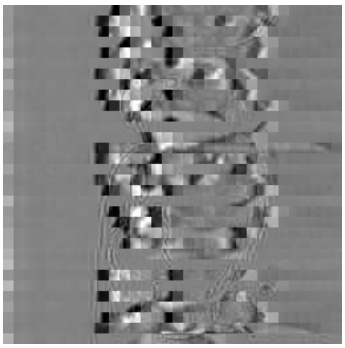
(e) 0.3



(f) 0.5



(g) 0.7



(h) 0.9

Figure 5-7: Bird image encrypted with different transform coefficients

5.3.4 Quality Loss on Post-Encryption Compression

There is a difference between compressed original image and recovered image from compressed encrypted image. This difference comes from the disharmony between out integer to integer transformation and rounding operation in the compression. There is no such difference without using our proposed integer to integer transformation. This difference has no importance, because both the compressed original image and compressed encrypted image has same PSNR values with respect to the original image. In this experiment this difference will be examined.

Proposed encryption method is applied to the original test images. The used parameters are shown on Table 5-5. The encrypted images are further compressed with QF 95, 90 … 35, 30 and reconstructed. The following values are recorded as shown Figure 5-8:

- The PSNR values between reconstructed image (I2) and compressed original image (I1) with same QF. It is named PSNR1 in the test.
- The PSNR values between original image (I0) and reconstructed image (I2). It is named PSNR2 in the test
- The PSNR values between original image (I0) and compressed original image (I1). It is named PSNR3 in the test
- The file size of compressed original image (I1) is measured in bit per pixel. It is named filesize1 in the test.
- The file size of reconstructed image (I2) is measured in bit per pixel. It is named filesize2 in the test.
- The file size of compressed encrypted image (I3) is measured in bit per pixel. It is named filesize3 in the test.

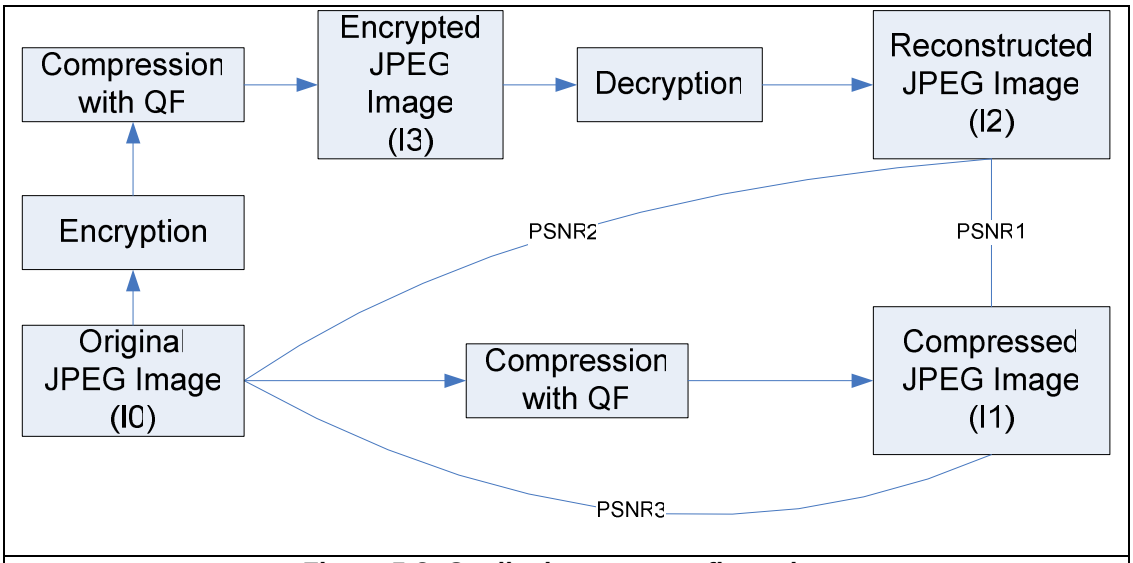


Figure 5-8: Quality loss test configuration

The important thing in this experiment is not the difference between I2 and I1, but the PSNR values of I2 and I1 with respect to I0. That means PSNR2/PSNR3 value should be 1. Table 5-6 shows the results for barb image and Table 5-7 shows the results for bird image. Figure 5-9 and Figure 5-10 shows the PSNR vs. QF plots. The results of the other images are in appendix A.

Table 5-5: Parameters for Experiment 5.3.5

Channels to apply transformation	Only 1. channel
Channels to apply scrambling	9(3x3) channels
Block Size	Fixed block size (1/8 or 1/16 times image width)
Transformation coefficient	0.5
Direction	Horizontal

Table 5-6: Quality Loss on Barb Image

QF	PSNR1 (dB)	PSNR2 (dB)	PSNR3 (dB)	PSNR2 over PSNR3	Filesize1 (bpp)	Filesize2 (bpp)	Filesize3 (bpp)
95	58.2042	43.7860	43.8298	0.9990	0.4061	0.4061	0.4069
90	56.9568	40.5198	40.5398	0.9995	0.2785	0.2785	0.2802
85	54.5360	38.6086	38.6495	0.9989	0.2206	0.2205	0.2228
80	53.1891	37.2518	37.3133	0.9984	0.1915	0.1915	0.1941
75	51.5710	36.1362	36.2233	0.9976	0.1676	0.1675	0.1710
70	49.5312	35.2638	35.3659	0.9971	0.1535	0.1535	0.1568
65	48.6710	34.5163	34.6079	0.9974	0.1402	0.1403	0.1439
60	47.2655	33.8430	33.9550	0.9967	0.1293	0.1293	0.1328
55	46.4612	33.2454	33.3811	0.9959	0.1210	0.1211	0.1249
50	45.7960	32.7011	32.8459	0.9956	0.1144	0.1145	0.1181
45	44.3044	32.1632	32.3343	0.9947	0.1073	0.1073	0.1110
40	43.6154	31.5581	31.7317	0.9945	0.0997	0.0998	0.1032
35	42.5100	30.8865	31.0772	0.9939	0.0921	0.0921	0.0955
30	41.1023	30.0733	30.2836	0.9931	0.0835	0.0835	0.0866

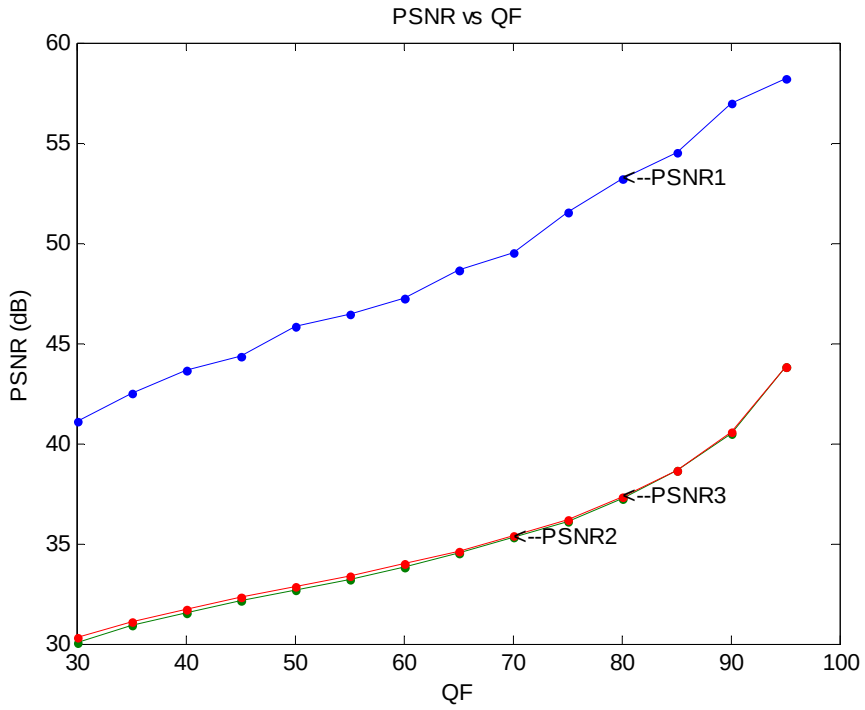


Figure 5-9: PSNR vs. QF for barb image

Table 5-7: Quality Loss on Bird Image

QF	PSNR1 (dB)	PSNR2 (dB)	PSNR3 (dB)	PSNR2 over PSNR3	Filesize1 (bpp)	Filesize2 (bpp)	Filesize3 (bpp)
95	58.3717	45.5516	45.6114	0.9987	0.2876	0.2876	0.2890
90	57.5473	43.3541	43.3885	0.9992	0.1840	0.1840	0.1869
85	54.6177	41.9800	42.0683	0.9979	0.1401	0.1400	0.1439
80	53.4227	40.9676	41.1143	0.9964	0.1180	0.1181	0.1230
75	52.0013	40.1863	40.3952	0.9948	0.0992	0.0993	0.1056
70	49.5671	39.5671	39.8469	0.9930	0.0898	0.0899	0.0962
65	48.4758	39.0686	39.3367	0.9932	0.0799	0.0801	0.0865
60	47.4610	38.5503	38.8998	0.9910	0.0738	0.0737	0.0801
55	47.0478	38.1566	38.5237	0.9905	0.0681	0.0681	0.0745
50	45.2584	37.5978	38.1776	0.9848	0.0651	0.0651	0.0710
45	44.7713	37.2518	37.8100	0.9852	0.0609	0.0610	0.0665
40	42.7551	36.5699	37.3901	0.9781	0.0568	0.0568	0.0620
35	42.6514	36.2991	36.9973	0.9811	0.0528	0.0527	0.0572
30	40.3569	35.3681	36.4609	0.9700	0.0489	0.0490	0.0527

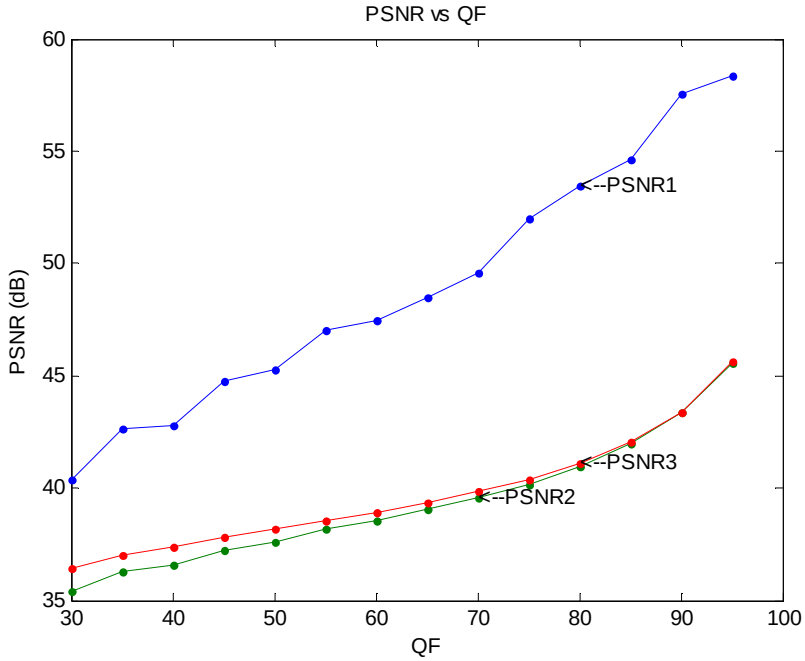


Figure 5-10: PSNR vs. QF for bird image

According to tables the difference has no importance, because the PSNR2/PSNR3 value is almost 1. And figures also show how close PSNR2 and PSNR3 are. It decreases when the compression increases. The worst value 0.97, which won't result any large visual difference for compressed images with QF 30. Moreover filesize1 and filesize2 are mostly same and filesize3 is slightly different than them. That means recovered and originally compressed images have mostly same file size and encrypted image has slight bit overhead. As a conclusion, our proposed method is almost perfectly robust to the post-encryption compression with integer to integer transformation and it is perfectly robust without integer to integer transformation.

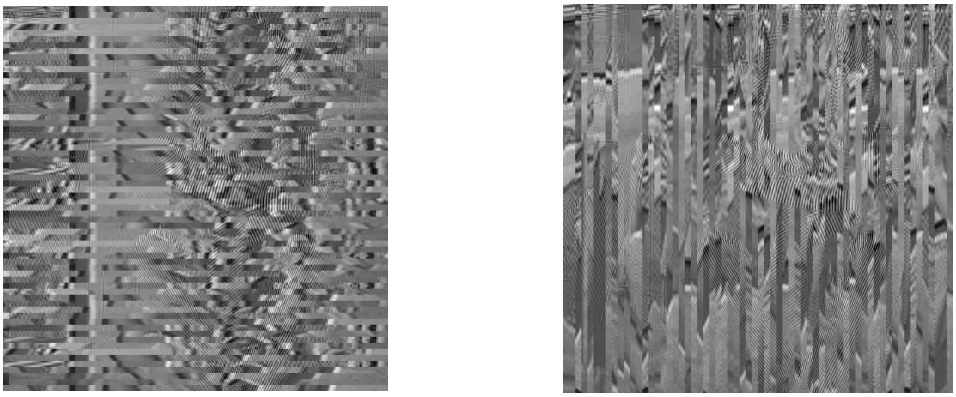
5.3.5 Vertical & Horizontal Modes

Every image has different visual characteristics with respect to vertical and horizontal directions. Same image can be seen different by human visual system if it is rotated 90^{o}. Visual scrambling quality of our proposed method is changing with respect to the harmony between image characteristics and direction of the applied method. In this experiment visual scrambling capability of our method on different directions is examined.

Proposed method is applied to the test images for each direction. The used parameters are shown on Table 5-8. Figure 5-11 and Figure 5-12 shows different characteristics of barb and zelda pictures. The results of the other images are in appendix A.

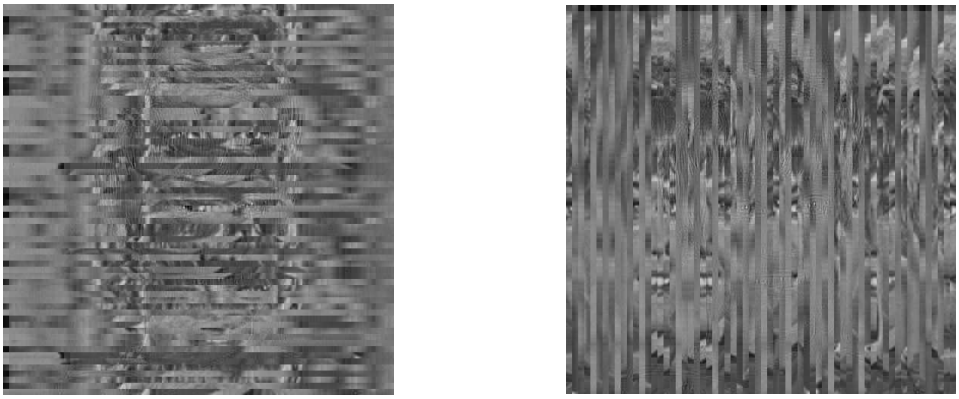
Table 5-8: Parameters for Experiment 5.3.5

Channels to apply transformation	Only 1. channel
Channels to apply scrambling	9(3x3) channels
Block Size	Fixed block size (1/8 times image width)
Transformation coefficient	0.5
Direction	Horizontal - Vertical



(a) Horizontal (b) Vertical

Figure 5-11: Barb image encrypted in different directions



(a) Horizontal (b) Vertical

Figure 5-12: Zelda image encrypted in different directions

Zelda image is not recognizable in the horizontal direction case, but it is recognizable in vertical direction case. On the other hand barb image is not recognizable in both directions. This shows that the encryption direction should be selected taking into consideration of the image. On the other hand, Figure 5-13 shows the encrypted Zelda image in the vertical direction with block size of 62, which is visually not recognizable. This shows that the block sizes and image dimensions should be relatively prime numbers.

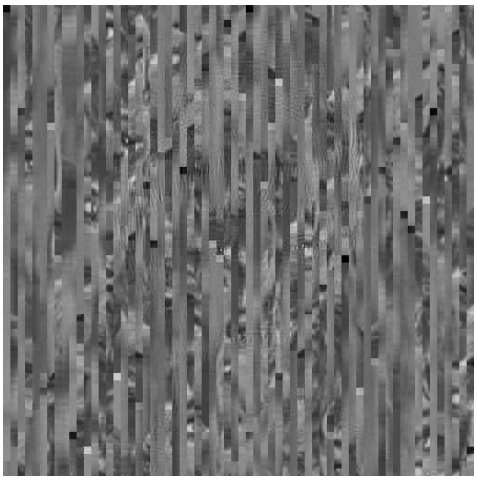


Figure 5-13: Zelda image encrypted in vertical mod with block size of 62.

5.4 Implementation on H.263

The usage of the proposed method on H.263 standard is implemented over H.263+ encoder/decoder coded by University of British Columbia. The decoder part of the codec is used to read/write H.263 bitstreams and the encoder part of the codec is used to encode encrypted I-frame.

Our encryption implementation inputs a H.263 bitstream and outputs encrypted H.263 bitstream. It is working in the following manner:

- While the decoder is reading the bitstream, decoder identifies the header values, coefficients and motion vectors.
- The decoder identification is used to select the data which is necessary for the encryption. The parts that is unnecessary for the encryption is written directly to output file.
- I-frame data read by the decoder is send to the encryption function which outputs encrypted I-frame. Encrypted I-frame is send to the I-frame encoder and it is written to the output file.
- The sign bits of the motion vectors and intracoded blocks are sent to the sign encryption function. Output of the sign encryption function sign bits are written to the output file.

The decryption implementation inputs an encrypted H.263 bitstream and outputs reconstructed H.263 bitstream. It is working same as encryption implementation.

5.5 Test Video Sequences

Video sequences that were commonly used in literature are selected for the experiments. The sequences are obtained in (or later converted to) H.263+ standard. Only a GOP portion of the sequences are used for tests, since every GOP will be treated same by the method.

Carphone Single object QCIF sequence with high motion foreground and background

Foreman Talking head QCIF sequence with high motion foreground and camera pan at the end.

Akiyo Almost still background and still foreground with motion QCIF sequence.

The sequences are included in the compact disc as H.263 files.

5.6 Experiments and Results on H.263

5.6.1 Encryption Performance

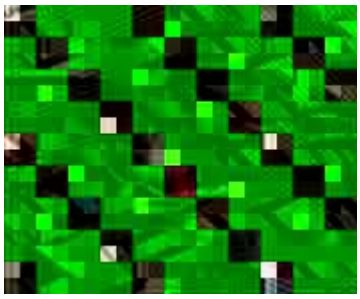
Our proposed method is applied on the I-frames of the H.263 sequences. Moreover intracoded blocks and the motion compensation are sign scrambled. In this experiment visual scrambling performance of our proposed method is examined.

The method mentioned in 4 is applied to the H.263 test sequences. The encryption parameters of our method are shown on Table 5-9. The original first frame and some of the encrypted frames of foreman sequence is shown on Figure 5-14. The results of the other sequences are in appendix A.

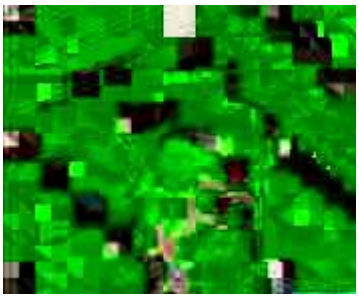
Table 5-9: Parameters for Experiment 5.6.1

	Luminance	Chrominance
DCT Channels	4x4	2x2
Transform Coefficient	0.9	0.9
Direction	Horizontal	Horizontal
Block Size	16	4

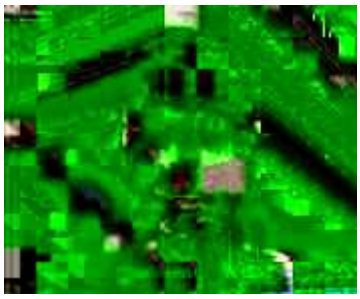
(a) Original 1. frame



(b) Encryped 1. frame

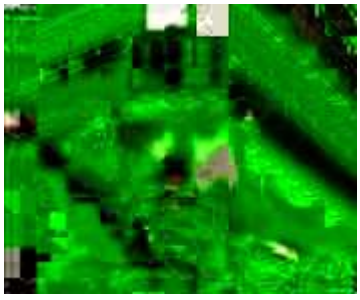


(c) Encrypted 5. frame

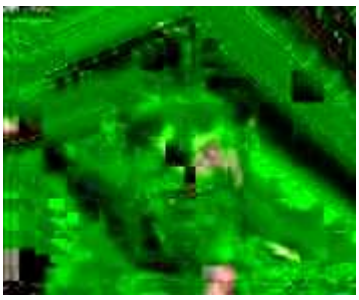


(d) Encrypted 9. frame

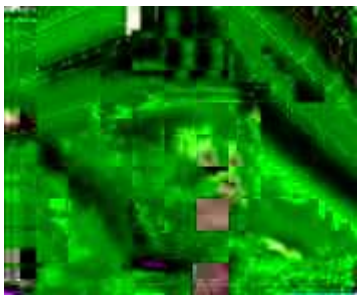
(e) Encrypted 13. frame



(f) Encrypted 17. frame



(g) Encrypted 21. frame



(h) Encrypted 25. frame

Figure 5-14: Original and encrypted frames of Foreman.

First frames are visually not recognizable. But at the 21 and 25 frame of foreman, his face is a little bit recognizable. This is expected, because of the visual scrambling leakage of the sign scrambling. Our proposed method is working well on H263 sequences, but it should be supported by a method for P-frames.

5.6.2 Effect of P-Frame Sign Scrambling

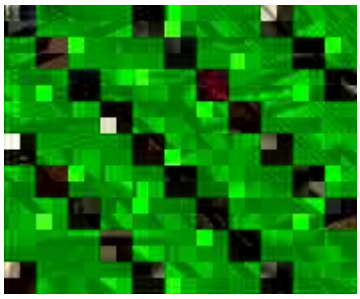
Our proposed method is applied on the I-frames of the H.263 sequences. Moreover intracoded blocks and the motion compensation are sign scrambled. In this experiment the effect of P-frame sign scrambling algorithm is examined.

The method mentioned in 4 is applied to the H.263 test sequences. The encryption parameters of our method are shown on Table 5-10. The original first frame and some of the encrypted frames of foreman sequence is shown on Figure 5-15.

Table 5-10: Parameters for Experiment 5.6.2

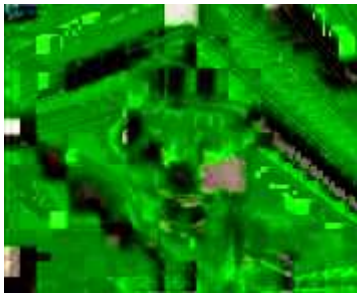
	Luminance	Chrominance
DCT Channels	4x4	2x2
Transform Coefficient	0.9	0.9
Direction	Horizontal	Horizontal
Block Size	16	4

(a) Original 1. frame



(b) Encryped 1. frame

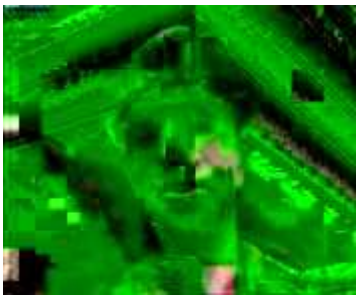
(c) Encrypted 5. frame



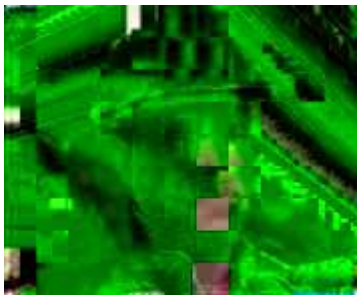
(d) Encrypted 9. frame

(e) Encrypted 13. frame

(f) Encrypted 17. frame



(g) Encrypted 21. frame



(h) Encrypted 25. frame

Figure 5-15: Original and encrypted frames of Foreman.

First frames are visually not recognizable like P-frame scrambled case. But at the 21 and 25 frame of foreman, his face is more recognizable than sign scrambled case. Because of the motion compensation and intracoded blocks, scrambled stream is recovering itself slowly frame by frame without sign scrambling. This slow recovery would possibly bring a security leakage. This is expected, our method is not secure without a supporting P-frame scrambling method. Even the simplest method used on P-frames will result a big difference in the security manner.

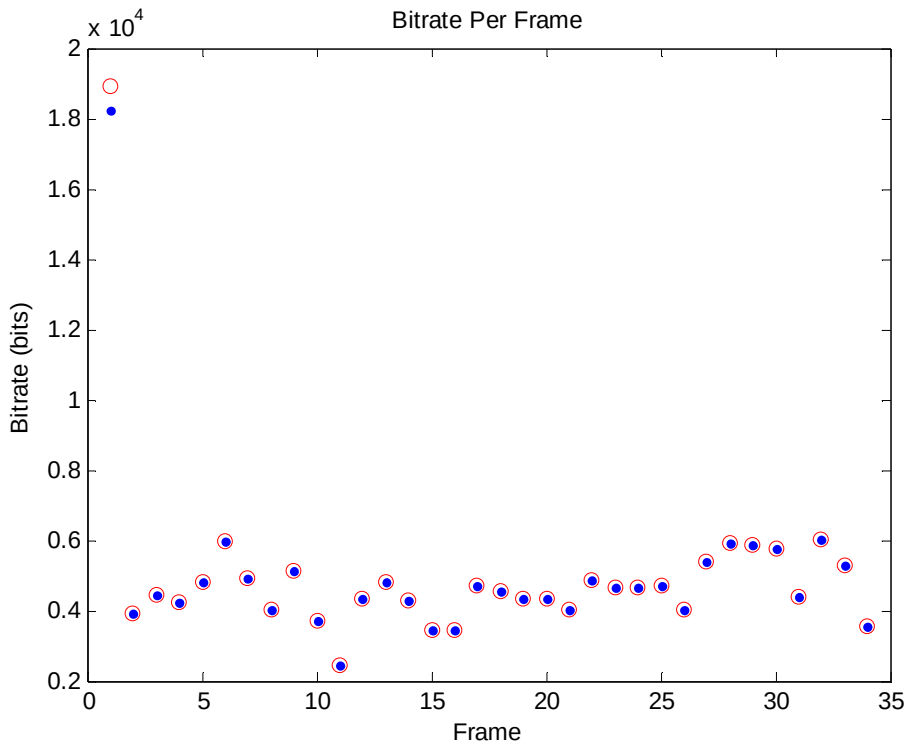
5.6.3 Effect on Bitrate

Our proposed method is applied on the I-frames of the H.263 sequences. Moreover intracoded blocks and the motion compensation are sign scrambled. In this experiment bitrate overhead of our proposed method is examined.

The method mentioned in Chapter 4 is applied to the H.263 test sequences. The encryption parameters of our method are shown on Table 5-11. The bitrate of the original foreman sequence and encrypted foreman sequence per frame are shown on Figure 5-16. And the bitrate difference between original and encrypted sequence per frame is shown on Figure 5-17.

Table 5-11: Parameters for Experiment 5.6.3

	Luminance	Chrominance
DCT Channels	4x4	2x2
Transform Coefficient	0.9	0.9
Direction	Horizontal	Horizontal
Block Size	16	4



"O": Encrypted "." : Original

Figure 5-16: Bitrates of original and encrypted sequence per frame

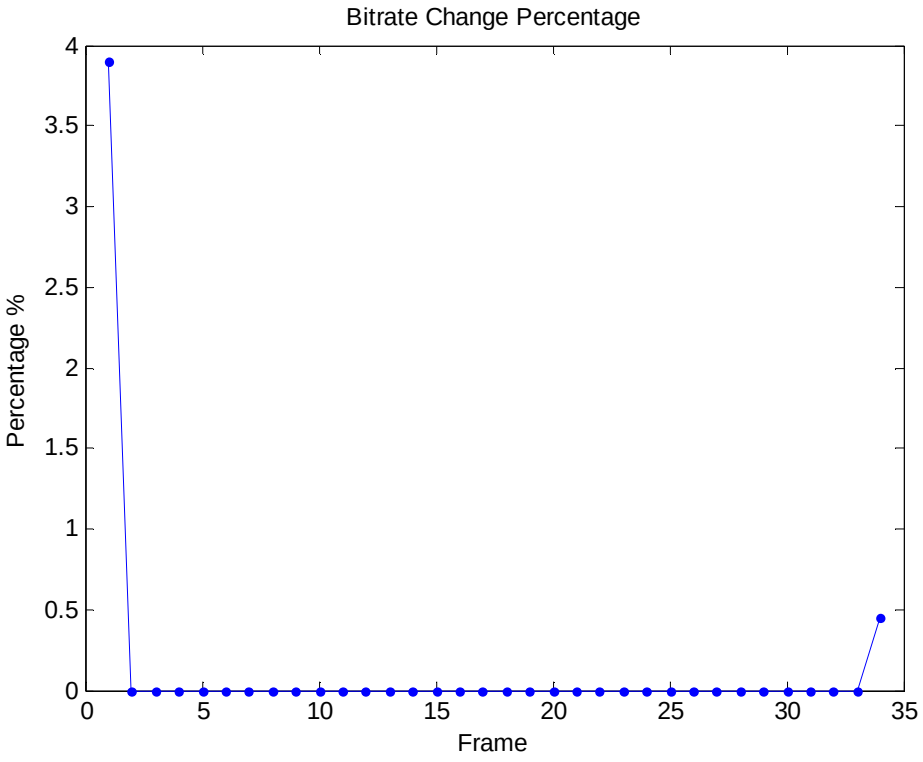


Figure 5-17: Bitrate change percentage per frame of Foreman

Only the scrambled I-frame and the last P-frame have bitrate change. And total bitrate overhead is 0.004%. This is expected, since our proposed method brings higher bitrate overheads on low resolutions, the I-frame has 4% bitrate change which is more than 1%. And the sign scrambling used in the method does not result any bitrate difference on the P-frames. Only the last frame has a slight bitrate change because of the bit stuffing used in H.263. Moreover overall bitrate change is nearly 0%. That means usage of our proposed method with sign scrambling on H.263 video sequences results slight bitrate overhead.

CHAPTER 6

CONCLUSION AND FUTURE WORK

New advances in technology allowed not only widely usage, but also uncontrollable distribution of multimedia content. This phenomenon made access control of the multimedia content essential in both commercial broadcasting and multimedia distribution networks. The attempts of encrypting multimedia data by classical methods developed for text messages such as AES or DES succeed in security, but fail in practice due to computational and bandwidth requirements. Therefore special multimedia encryption methods are required to establish security of distribution channels.

Designing an algorithm, specifically for multimedia data, which has low bitrate overhead due to encryption, maximum security and realizable computational complexity is not an easy task to accomplish. There are a number of different methods proposed for multimedia encryption in the literature, but these methods could not satisfy all the desired requirements. In this work a new method is proposed for multimedia data encryption, which is robust to the intermediate bitrate conversion operation and which has an adjustable security level and operates with slight bitrate overhead. The intermediate bitrate conversion operation allows the content owner to encrypt the data only once and leave the data in the channel. Depending on the communication channel constraints or the device capabilities of the end-users, the encrypted data can be compressed to a lower bitrate, i.e. transcoded; without decrypting the data with the proposed technique.

Table 6-1 compares the proposed method with other methods mentioned in Chapter 2.

Table 6-1: Comparison of Encryption Methods

	Sign Scrambling [1]	Slice Scrambling [7]	DCT Coefficient Scrambling [3]	Line Scrambling [9]	Proposed Method
Security	Low	Low	Medium	Medium	Medium High
Key Size	Small	Medium Large	Small	Medium	Medium
Overhead on Size	Small	Medium Large	Large	Huge	Medium
Robustness to Bitrate Conversion	Yes	Yes	No	No	Yes
Implementation Complexity	Low	Low	Low	Low	Medium
Algorithm Speed	High	High	High	High	Medium High
Reconstruction Error	None	None	None	None	None

6.1 Features of the Proposed Method

Unlike previously developed image encryption methods, the proposed method is robust to the post-encryption bitrate conversion and has acceptable security. That means, intentional bitrate adjustment made by the channel or unintentional bitrate adjustment made by unauthorized people will not affect the visual quality of the reconstructed image.

Figure 6-1 shows how our proposed method secures the multimedia transmission on public network. Unauthorized people could not access to the multimedia content. But the authorized users could access to the multimedia content even the encrypted image is compressed in the network.

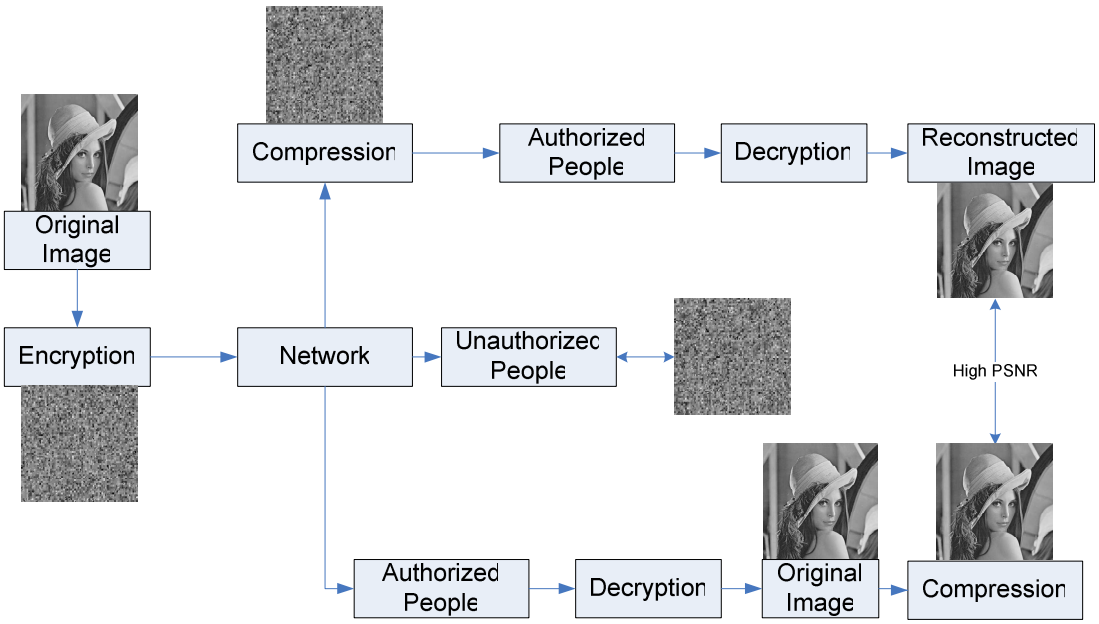


Figure 6-1: Secure Transmission

In addition, it provides selective complexity/security level with the adjustment of its parameters in the following manner:

- Increasing or decreasing number of encrypted DCT channels.
- Increasing or decreasing scrambling fixed block sizes.
- Using variable scrambling block sizes.
- Using cascade in vertical and horizontal directions.

The model is configurable with respect to the needs of the application by means of these parameters. And also it is open for further the modifications in the integer-to-integer transformation stage.

By experimentation, it is shown that proposed methods is robust to the post-encryption bitrate conversion as expected. There is a barely perceptible visual quality loss with the application of integer-to-integer transforms. But integer-to-integer transforms increase the security of the technique by decorrelating DCT channel coefficients. In addition, the cascade usage of the proposed method or selection of smaller block sizes can also be used to increase the security of the encryption. Moreover, the method has a slight bitrate increase typically smaller than 1%.

Apart from the still image encryption capability of the method, the experiments made on the H.263 codec show us the method is also suitable for the 8x8 DCT based video standards. Experiments show that even the 25th frame after the encrypted I-frame can not be recognized with ease.

6.2 Main Drawbacks

It is difficult to compare encryption results, since there is no commonly accepted and used test setup in the field.

The integer-to-integer transformation, proposed in this work, is applicable but not optimal transformation for erasing the visual characteristics hidden on the DCT channels. It introduces a quality loss on the transcoded image at the benefit of increasing the security level. One could replace it with more appropriate transformation, but it is not an easy task to accomplish because of the difficulties in designing integer-to-integer decorrelating transforms.

The proposed method is applied to I-frames of H.263 sequences to provide security. In this work, also sign encryption algorithm is applied to secure P-frames and motion vector information of video sequences. A more comprehensive technique should be designed to secure P-frames and motion vectors.

6.3 Suggested Future Work

The following subjects are suggested for further study, which can make use of findings of this thesis:

- The security performance of the proposed method can be tested with chipper text attacks.
- The decorrelating performance of the integer-to-integer transformation can be increased. Another integer to integer transforms with better decorrelating capability and little computational overhead could be used instead of the proposed transformation.
- Other P-frame encryption algorithms can be developed.
- The proposed method can be applied to MPEG-2. The adaptation of the proposed method to MPEG-2 should not be very difficult, because of the similar nature of the MPEG-2 and H.263.

REFERENCES

[1] C. Shi and B. Bhargava, “A fast MPEG video encryption algorithm,” in Proc. 6th ACM Int. Conf. Multimedia, Sep 1998.

[2] C. Shi and B. Bhargava, “MPEG video encryption in real-time usinf secret key cryptography,” Proc. PDPTA’99, vol. 6, p. 2822, Jun.1999.

[3] L. Tang, “Methods for encrypting and decrypting MPEG video data efficiently,” in Proc. 4th ACM Int. Multimedia Conf, 1996, pp. 219-229.

[4] L. Qiao and K. Nahrstedt, “Comparison of MPEG encryption algorithms,” International Journal on Computer and Graphics, Special Issue on Data Security in Image Communications and Network, VOL. 22, no. 3, 1998.

[5] Chung-Ping Wu and C.-C. Jay Kuo, “Design of Integrated Multimedia Compression and Encryption Systems,” IEEE Transactions on Multimedia, Vol. 7, No. 5, 2005

[6] M. Kıvanç Mıhçak and Ramarathnam Venkatesan, “A Perceptual Audio Hashing Algorithm: A tool For Robust Audio Identification and Information Hiding,” presented at the 4th Int. Information Hiding Workshop, Apr. 2001.

[7] Wenjun Zeng, “Efficient Frequency Domain Selective Scrambling of Digital Video,” IEEE Transactions on Multimedia, VOL. 5, No. 1, 2003

[8] G.L. Hobbs, “Video Scrambling,” U.S. Patent 5 815 572, Sept. 29, 1998.

[9] D. Zeidler and J. Griffin, “Method and Apparatus for Television Signal Scrambling Using Block Shuffling,” U.S. Patent 5 321 748, June 14, 1994.

[10] ITU-T Recommendation H.263 Version 2 (H.263+), “Video Coding for Low Bitrate Communication,” 1998

[11] Vivek K. Goyal, “Transform Coding with Integer-to-Integer Transforms,” IEEE Transactions on Information Theory, VOL. 46, No.2, March 2000

[12] J. Daemen and V. Rijmen, “AES Proposal: Rijndael,” AES Algorithm Submission, September 3, 1999

[13] Y. Mao and M. Wu, “A Joint Signal Processing and Cryptographic Approach to Multimedia Encryption,” IEEE Transactions on Image Processing, VOL. 15, No. 7, July 2006

[14] A. Oppenheim, J. Buck, R Schafer, “Discerete-Time Signal Processing, 2nd edition,” Prentice-Hall, 1998

[15] IEEE Transactions on Circuits and Systems for Video Technology: Special Issue on Authentication, Copyright Protection, and Information Hiding, Vol. 13, No. 8, August 2003.

[16] Howard Cheng and Xiaobo Li, “Partial Encryption of compressed images and videos,” IEEE Transactions on Signal Processing, Vol. 48, no. 8, pp. 2439-2451, 2000.

[17] Rafael C. Gonzalez, Richard E. Woods, “Digital Image Processing, 2nd edition,” Addison-Wesley, 1993.

[18] Jae S. Lim, “Two-Dimensional Signal and Image Processing,” Prentice-Hall, 1990.

[19]] ISO/IEC IS 109818-1 (ITU-T Rec.T.81), “Digital Compression and Coding of Continuous-tone Still Images,” 1992.

[20] X. Liu and A.M. Eskicioglu “Selective Encryption of Multimedia Content in Distribution Networks: Challenges and New Directions,” IASTED International Conference on Communications, Internet and Information Technology (CIIT 2003), Scottsdale, AZ, November 17-19, 2003.

[21] M. Van Droogenbroeck and R. Benedett, “Techniquies for a Selective Encryption of Uncompressed and Compressed Images,” Proceedings of Advanced Concepts for Intelligent Vision Systems (ACIVS) 2002, Ghent, Belgium, September 9-11, 2002.

[22] M. Podesser, H-P. Schmidt and A. Uhl, “ Selective Bitplave Encryption for Secure Transmission of Image Data in Mobile Environments,” 5th Nordic Signal Processing Symposium, on board Hurtigruten, Norway, October 4-7, 2002.

[23] J. Meyer and F. Gadegast, “Security Mechanisms for Multimedia Data with Example MPEG-1 Video,” Project Description on SECMPEG, Technical University of Berlin, Germany, May 1995.

[24] G. A. Spanos and T.B. Mapled, “Security for Real-Time MPEG Compressed Video in Distributed Multimedia Applications,” in Conference on Computers and Communications, pp. 72-78, 1996.

[25] A. M. Alattar, G. I. Al-Regib and S.A. Al-Semari, “Improved Selective Encryption techniques for Secure Transmission of MPEG Video Bit-Streams,”

Proceedings of the 1999 International Conference on Image Processing (ICIP ‘99), Vol. 4, Kobe, Japan, October 24-28, pp. 256-260, 1999.

[26] S. U. Shin, K. S. Sim and K. H. Rhee, “A Secrecy Scheme for MPEG Video Data Using the Joint of Compression and Encryption,” Second International Workshop on Information Security (ISW ‘99), Kuala Lumpur, Malaysia, November 1999, Lecture Notes in Computer Science, Vol. 1729, pp. 191-201, 1999.

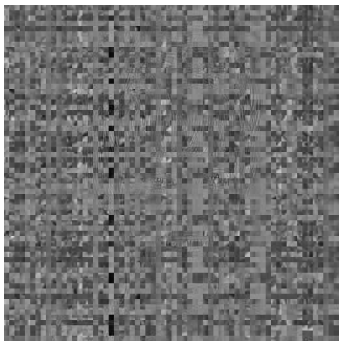
[27] J. Wen, M. Severa, W. Zeng, M. Luttrell, and W. Jin, “A Format-Compliant Configurable Encryption Framework for Access Control of Video,” IEEE Transactions of Circuits and Systems for Video Technology, Vol. 12, No. 6, pp. 545-557, June 2002.

[28] M. Wu and Y. Mao, “Communication-Friendly Encryption of Multimedia,” 2002 International Workshop on Multimedia Signal Processing, St. Thomas, US Virgin Islands, December 9-11, 2002.

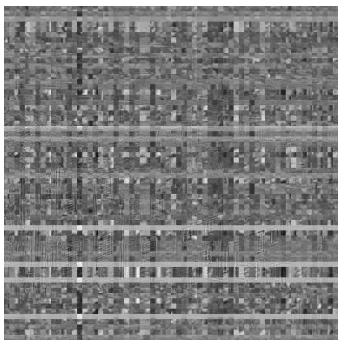
[29] C.-P. Wu and C.-C. J. Kuo, “Efficient Multimedia Encryption via Entropy Codec Design,” Proceedings of SPIE Security and Watermarking of Multimedia Content III, Volume 4314, San Jose, CA, January 2001.

APPENDICES

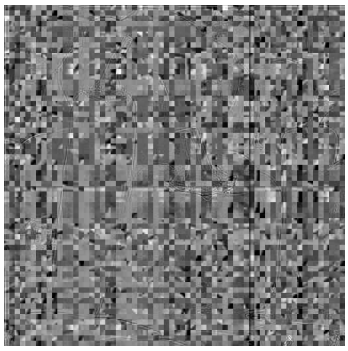
APPENDIX A



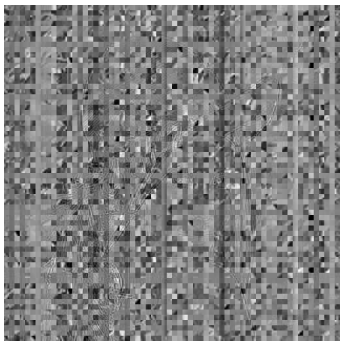
(a) zelda



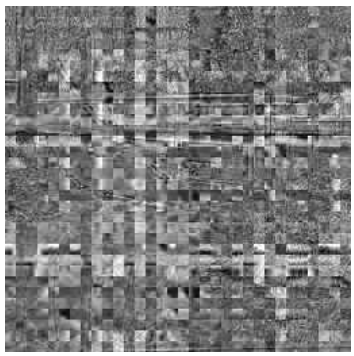
(b) goldhill



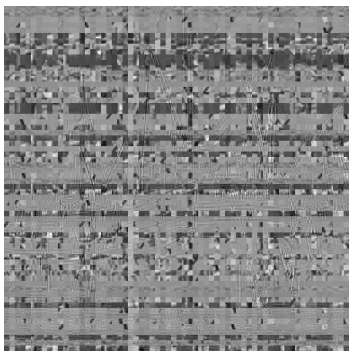
(c) peppers



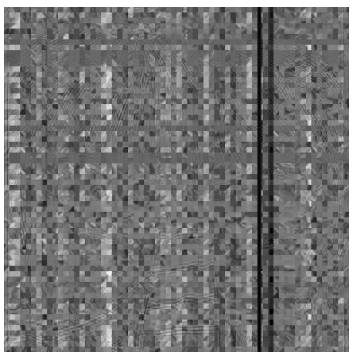
(d) lena



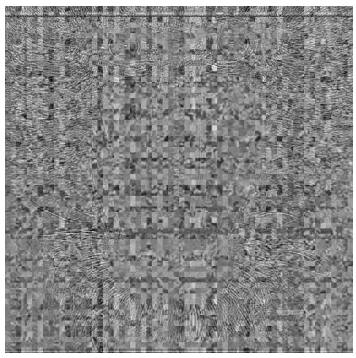
(d) bridge



(e) boat



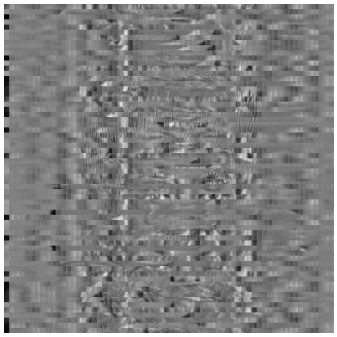
(f) girls



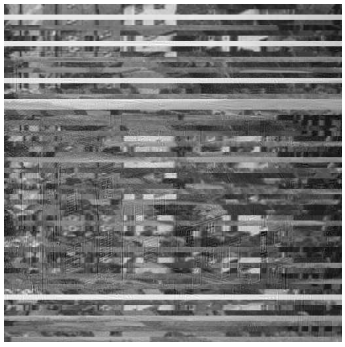
(g) mandrill

Figure A-1: Results of 5.3.1

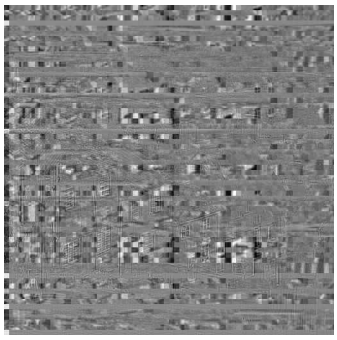
(a) zelda (0)



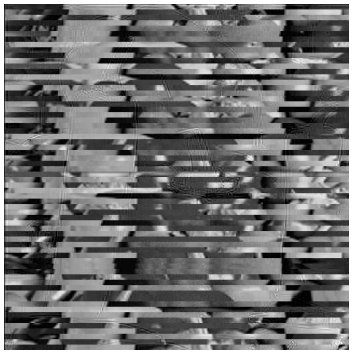
(b) zelda (0.9)



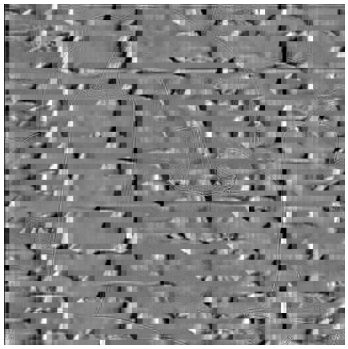
(d) goldhill (0)



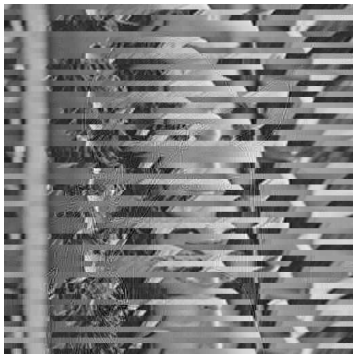
(e) goldhill (0.9)



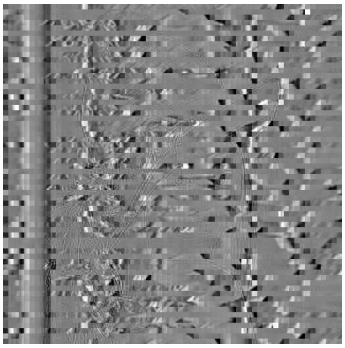
(f) peppers (0)



(g) peppers (0.9)

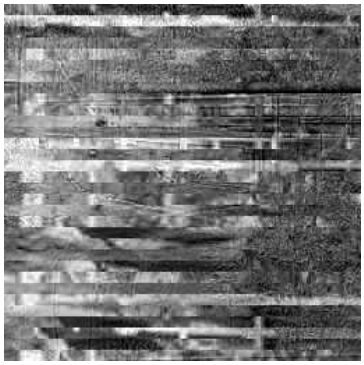


(h) lena (0)

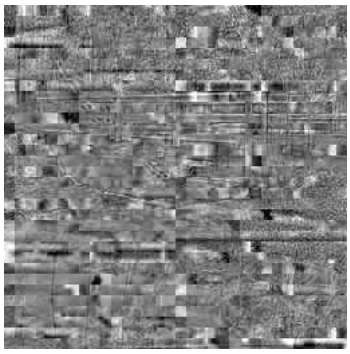


(i) lena (0.9)

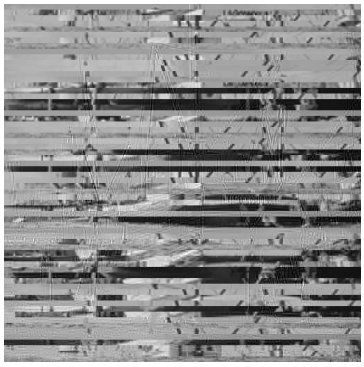
Figure A-2: Results of 5.3.2



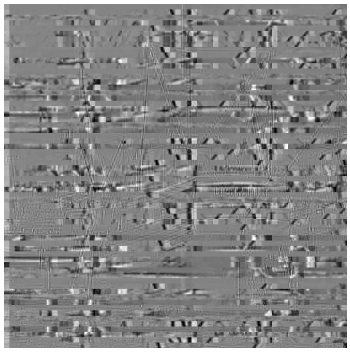
(j) bridge (0)



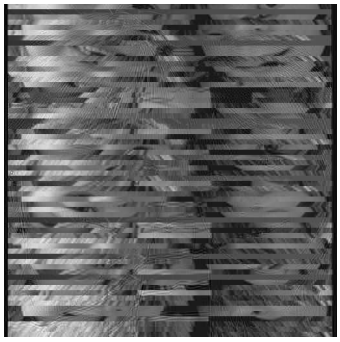
(k) bridge (0.9)



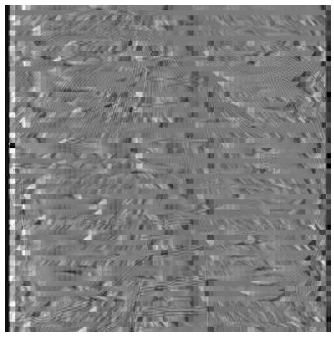
(l) boat (0)



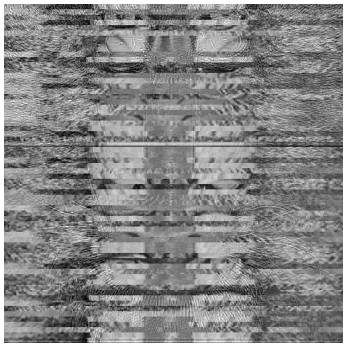
(m) boat (0.9)



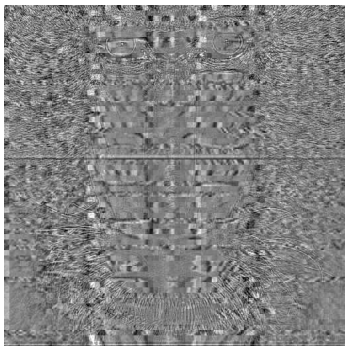
(n) girls (0)



(o) girls (0.9)



(p) mandrill (0)



(r) mandrill (0.9)

Figure A.2 (cont'd)

Table A-1: Results of 5.3.4 for Zelda Image

QF	PSNR1	PSNR2	PSNR3	PSNR2 over PSNR3	Filesize1	Filesize2	Filesize3
95	58.3368	43.7752	43.8146	0.9991	0.3306	0.3306	0.3312
90	57.2556	41.5732	41.5949	0.9995	0.2048	0.2047	0.2058
85	54.3509	40.4809	40.5539	0.9982	0.1489	0.1489	0.1503
80	53.1604	39.7424	39.8542	0.9972	0.1253	0.1252	0.1270
75	51.9525	39.1624	39.3344	0.9956	0.1055	0.1055	0.1078
70	49.5759	38.6911	38.9187	0.9942	0.0958	0.0958	0.0982
65	48.8887	38.3403	38.5527	0.9945	0.0851	0.0852	0.0878
60	47.4437	37.9274	38.2139	0.9925	0.0773	0.0773	0.0800
55	46.8746	37.5905	37.9184	0.9914	0.0713	0.0713	0.0742
50	45.9217	37.2386	37.6376	0.9894	0.0676	0.0676	0.0703
45	44.5452	36.8410	37.3598	0.9861	0.0627	0.0627	0.0654
40	43.7815	36.4781	37.0089	0.9857	0.0575	0.0575	0.0600
35	42.6794	36.0266	36.6554	0.9828	0.0532	0.0531	0.0555
30	41.1183	35.4114	36.1931	0.9784	0.0482	0.0481	0.0502

Table A-2: Results of 5.3.4 for Goldhill Image

QF	PSNR1	PSNR2	PSNR3	PSNR2 over PSNR3	Filesize1	Filesize2	Filesize3
95	58.4292	42.8175	42.8543	0.9991	0.4308	0.4318	0.4331
90	57.1369	39.1761	39.1902	0.9996	0.2875	0.2877	0.2896
85	54.5935	37.4613	37.4939	0.9991	0.2217	0.2209	0.2232
80	53.2292	36.3671	36.4173	0.9986	0.1894	0.1882	0.1908
75	51.9042	35.5672	35.6433	0.9979	0.1644	0.1625	0.1653
70	49.7608	35.0029	35.0984	0.9973	0.1492	0.1470	0.1497
65	48.7541	34.5158	34.6081	0.9973	0.1353	0.1327	0.1358
60	47.3588	34.0789	34.1985	0.9965	0.1247	0.1218	0.1247
55	46.6962	33.6930	33.8373	0.9957	0.1163	0.1133	0.1162
50	45.7811	33.3489	33.5237	0.9948	0.1098	0.1065	0.1094
45	44.5258	33.0062	33.2077	0.9939	0.1024	0.0989	0.1018
40	43.8860	32.6361	32.8539	0.9934	0.0947	0.0910	0.0939
35	42.7129	32.2550	32.5045	0.9923	0.0870	0.0831	0.0860
30	41.0171	31.7203	32.0563	0.9895	0.0787	0.0744	0.0772

Table A-3: Results of 5.3.4 for Peppers Image

QF	PSNR1	PSNR2	PSNR3	PSNR2 over PSNR3	Filesize1	Filesize2	Filesize3
95	58.8075	42.6239	42.6480	0.9994	0.4013	0.4013	0.4020
90	57.4753	39.0043	39.0156	0.9997	0.2575	0.2574	0.2586
85	54.7740	37.6229	37.6531	0.9992	0.1894	0.1894	0.1913
80	53.4420	36.8534	36.9048	0.9986	0.1586	0.1585	0.1608
75	52.0601	36.2948	36.3737	0.9978	0.1354	0.1353	0.1383
70	49.6054	35.8699	35.9872	0.9967	0.1226	0.1226	0.1257
65	48.9580	35.5253	35.6265	0.9972	0.1096	0.1096	0.1130
60	47.5356	35.1670	35.3063	0.9961	0.1005	0.1005	0.1042
55	47.0931	34.8574	35.0241	0.9952	0.0930	0.0930	0.0970
50	46.0274	34.5722	34.7774	0.9941	0.0879	0.0879	0.0917
45	44.7843	34.2676	34.5232	0.9926	0.0817	0.0817	0.0856
40	43.7895	33.9426	34.2326	0.9915	0.0755	0.0755	0.0794
35	42.9371	33.6218	33.9367	0.9907	0.0702	0.0702	0.0738
30	41.7055	33.1657	33.5505	0.9885	0.0641	0.0641	0.0675

Table A-4: Results of 5.3.4 for Lena Image

QF	PSNR1	PSNR2	PSNR3	PSNR2 over PSNR3	Filesize1	Filesize2	Filesize3
95	58.7228	42.2183	42.2447	0.9994	0.4211	0.4210	0.4223
90	57.4627	38.3764	38.3867	0.9997	0.2734	0.2734	0.2751
85	54.6293	37.0276	37.0557	0.9992	0.1962	0.1962	0.1983
80	53.5805	36.2818	36.3253	0.9988	0.1612	0.1612	0.1638
75	52.1566	35.7328	35.8025	0.9981	0.1351	0.1352	0.1385
70	49.9119	35.3324	35.4255	0.9974	0.1208	0.1208	0.1243
65	49.0026	34.9946	35.0870	0.9974	0.1071	0.1071	0.1114
60	47.7049	34.6684	34.7896	0.9965	0.0968	0.0969	0.1012
55	46.9787	34.3800	34.5329	0.9956	0.0892	0.0893	0.0938
50	46.0353	34.1154	34.3011	0.9946	0.0834	0.0834	0.0880
45	44.8286	33.8325	34.0654	0.9932	0.0772	0.0773	0.0818
40	44.0797	33.5553	33.7920	0.9930	0.0710	0.0710	0.0755
35	42.7590	33.1935	33.5045	0.9907	0.0653	0.0652	0.0694
30	41.5266	32.7619	33.1276	0.9890	0.0589	0.0589	0.0628

Table A-5: Results of 5.3.4 for Bridge Image

QF	PSNR1	PSNR2	PSNR3	PSNR2 over PSNR3	Filesize1	Filesize2	Filesize3
95	59.0965	42.2940	42.3205	0.9994	0.5894	0.5913	0.5924
90	57.2070	37.1721	37.1781	0.9998	0.4345	0.4351	0.4359
85	54.4732	34.4295	34.4460	0.9995	0.3517	0.3505	0.3513
80	53.2563	32.7812	32.8020	0.9994	0.3026	0.3006	0.3017
75	52.0068	31.6242	31.6504	0.9992	0.2660	0.2631	0.2643
70	49.6121	30.8592	30.8971	0.9988	0.2414	0.2377	0.2390
65	49.3022	30.2480	30.2771	0.9990	0.2198	0.2153	0.2166
60	47.6949	29.7493	29.7899	0.9986	0.2030	0.1983	0.1995
55	46.9925	29.3330	29.3810	0.9984	0.1893	0.1843	0.1857
50	46.1228	28.9942	29.0484	0.9981	0.1785	0.1731	0.1745
45	44.7586	28.6598	28.7240	0.9978	0.1679	0.1618	0.1632
40	43.6386	28.2806	28.3583	0.9973	0.1566	0.1503	0.1517
35	42.7912	27.9164	27.9973	0.9971	0.1446	0.1381	0.1394
30	41.3255	27.4876	27.5853	0.9965	0.1325	0.1253	0.1265

Table A-6: Results of 5.3.4 for Boat Image

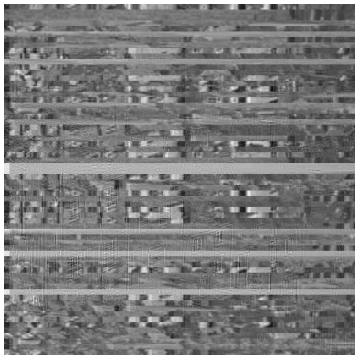
QF	PSNR1	PSNR2	PSNR3	PSNR2 over PSNR3	Filesize1	Filesize2	Filesize3
95	58.2012	43.7684	43.8143	0.9990	0.3730	0.3730	0.3746
90	57.4402	40.7903	40.8152	0.9994	0.2473	0.2472	0.2510
85	54.5373	39.1227	39.1693	0.9988	0.1924	0.1924	0.1975
80	53.1206	37.9723	38.0509	0.9979	0.1645	0.1646	0.1705
75	51.7478	37.0741	37.1872	0.9970	0.1425	0.1425	0.1495
70	49.3771	36.3966	36.5461	0.9959	0.1305	0.1304	0.1373
65	48.8420	35.8450	35.9669	0.9966	0.1187	0.1187	0.1257
60	47.5913	35.3255	35.4755	0.9958	0.1098	0.1098	0.1165
55	46.5976	34.8356	35.0390	0.9942	0.1028	0.1027	0.1096
50	45.5504	34.4129	34.6546	0.9930	0.0974	0.0975	0.1039
45	44.8443	34.0188	34.2661	0.9928	0.0917	0.0917	0.0979
40	43.4626	33.5014	33.8100	0.9909	0.0856	0.0857	0.0915
35	42.6621	33.0466	33.3564	0.9907	0.0798	0.0798	0.0851
30	41.0407	32.4148	32.7898	0.9886	0.0732	0.0732	0.0781

Table A-7: Results of 5.3.4 for Girls Image

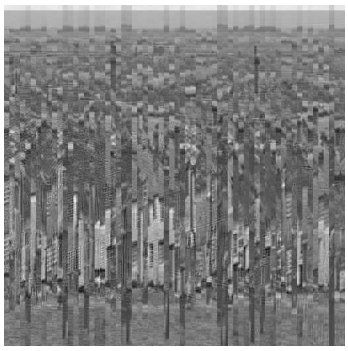
QF	PSNR1	PSNR2	PSNR3	PSNR2 over PSNR3	Filesize1	Filesize2	Filesize3
95	58.3062	45.0075	45.0654	0.9987	0.3774	0.3772	0.3790
90	57.2857	40.4518	40.4736	0.9995	0.2480	0.2480	0.2503
85	54.5994	38.6451	38.6874	0.9989	0.1894	0.1894	0.1924
80	52.9035	37.4881	37.5632	0.9980	0.1608	0.1609	0.1641
75	51.6944	36.6521	36.7557	0.9972	0.1383	0.1383	0.1425
70	49.1124	36.0459	36.1914	0.9960	0.1251	0.1251	0.1294
65	48.7454	35.5857	35.7024	0.9967	0.1137	0.1137	0.1184
60	47.1848	35.1301	35.2906	0.9955	0.1044	0.1044	0.1090
55	46.3875	34.7344	34.9341	0.9943	0.0966	0.0966	0.1013
50	45.6594	34.3845	34.6335	0.9928	0.0914	0.0914	0.0958
45	44.3632	34.0742	34.3357	0.9924	0.0854	0.0854	0.0898
40	43.4183	33.6751	33.9902	0.9907	0.0791	0.0791	0.0832
35	42.3691	33.2906	33.6409	0.9896	0.0733	0.0733	0.0771
30	40.8240	32.7396	33.1968	0.9862	0.0671	0.0671	0.0704

Table A-8: Results of 5.3.4 for Mandrill Image

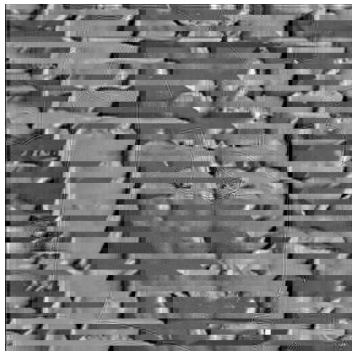
QF	PSNR1	PSNR2	PSNR3	PSNR2 over PSNR3	Filesize1	Filesize2	Filesize3
95	58.7296	42.1690	42.1939	0.9994	0.5972	0.5974	0.5977
90	57.3548	37.0172	37.0274	0.9997	0.4385	0.4384	0.4390
85	54.8483	34.2838	34.2979	0.9996	0.3548	0.3548	0.3555
80	53.4464	32.5353	32.5526	0.9995	0.3052	0.3052	0.3059
75	52.2646	31.2845	31.3081	0.9992	0.2680	0.2682	0.2691
70	49.7829	30.4133	30.4441	0.9990	0.2440	0.2440	0.2450
65	48.9431	29.6885	29.7152	0.9991	0.2223	0.2224	0.2236
60	47.5834	29.0881	29.1224	0.9988	0.2055	0.2054	0.2069
55	47.1689	28.5857	28.6226	0.9987	0.1919	0.1920	0.1935
50	45.9373	28.1588	28.2058	0.9983	0.1806	0.1806	0.1822
45	44.6263	27.7491	27.8040	0.9980	0.1695	0.1695	0.1714
40	43.8849	27.3002	27.3594	0.9978	0.1572	0.1572	0.1593
35	42.8283	26.8462	26.9136	0.9975	0.1450	0.1451	0.1474
30	41.3694	26.3371	26.4172	0.9970	0.1317	0.1317	0.1343



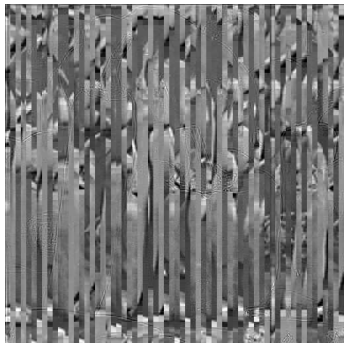
(a) goldhill horizontal



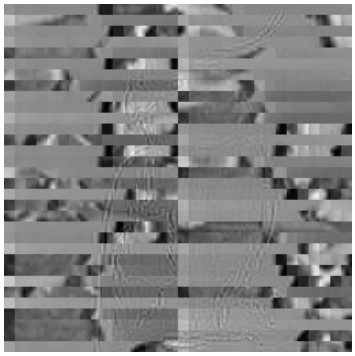
(b) goldhill vertical



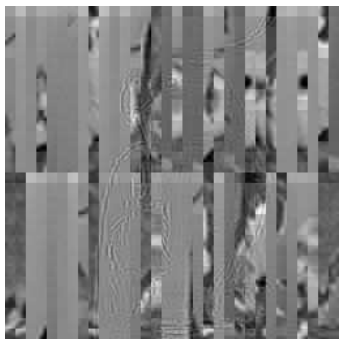
(c) peppers horizontal



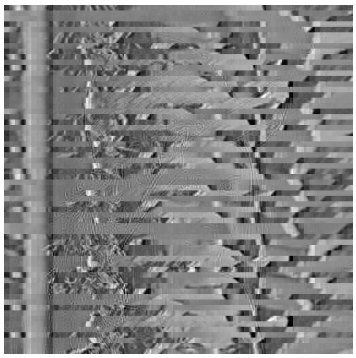
(d) peppers vertical



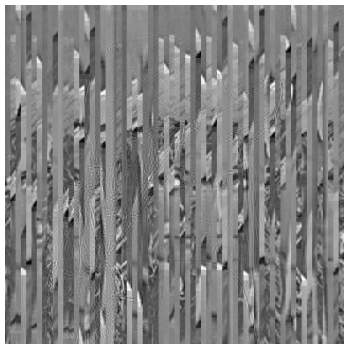
(e) bird horizontal



(g) bird vertical

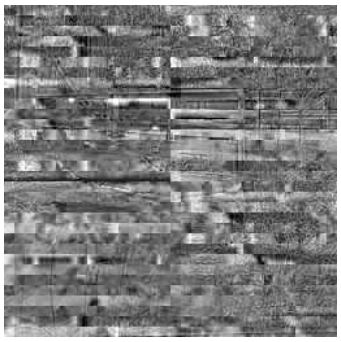


(h) lena horizontal

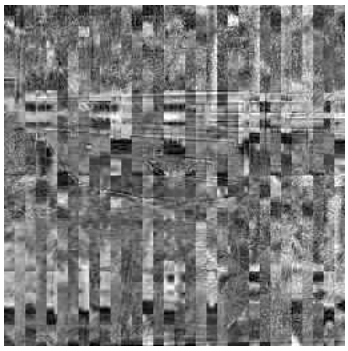


(i) lena vertical

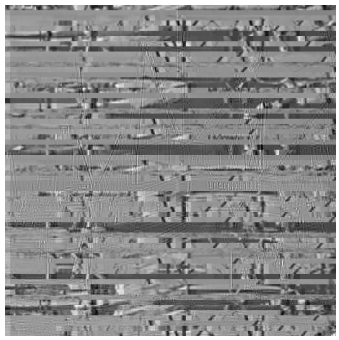
Figure A-3: Results of 5.3.4



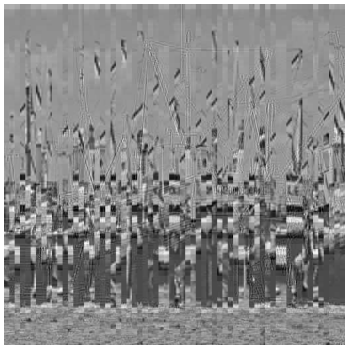
(j) bridge horizontal



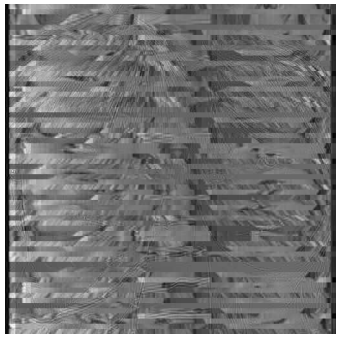
(k) bridge vertical



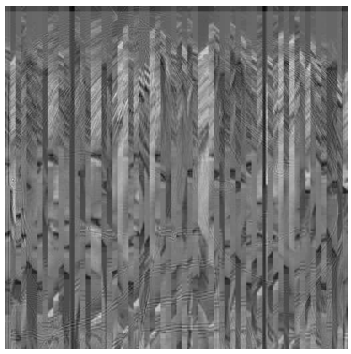
(l) boat horizontal



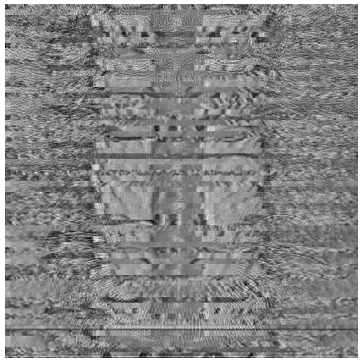
(m) boat vertical



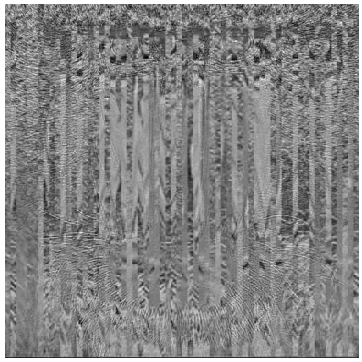
(n) girls horizontal



(o) girls vertical



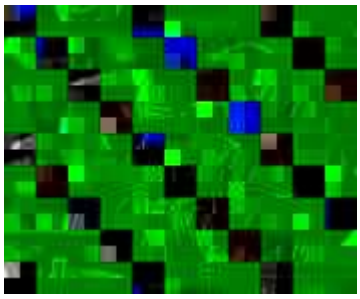
(p) mandrill horizontal



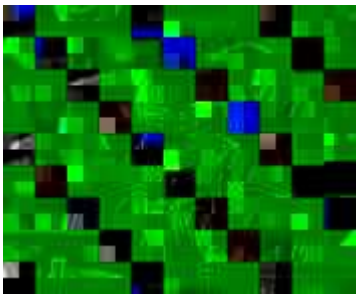
(r) mandrill vertical

Figure A.3 (cont'd)

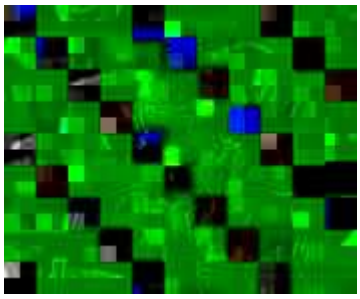
(a) Original 1. frame



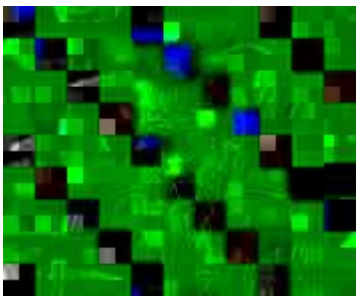
(b) Encryped 1. frame



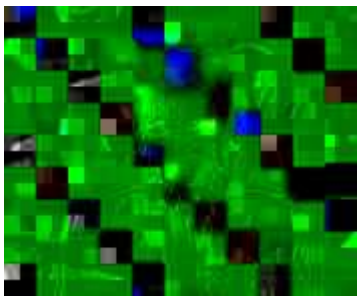
(c) Encrypted 5. frame



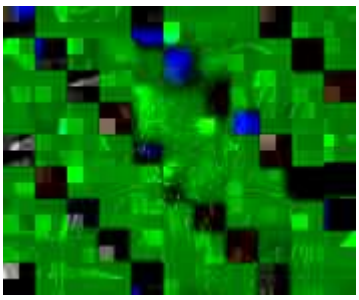
(d) Encrypted 9. frame



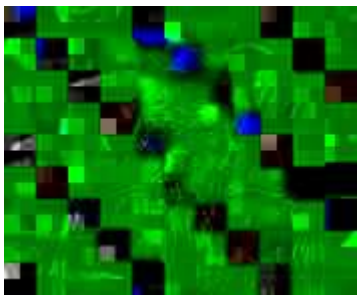
(e) Encrypted 13. frame



(f) Encrypted 17. frame

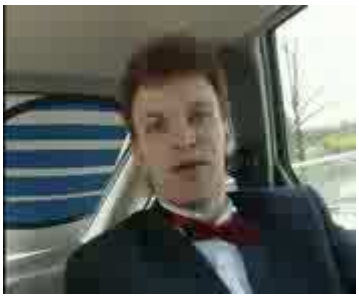


(g) Encrypted 21. frame

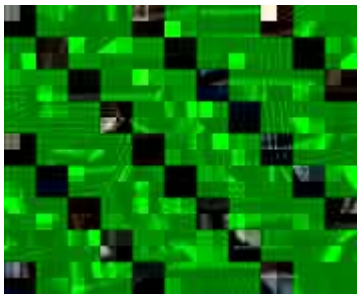


(h) Encrypted 25. frame

Figure A-4: Original and encrypted frames of Akiyo.

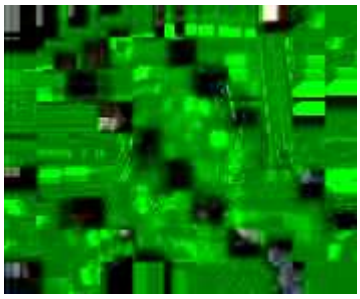


(a) Original 1. frame



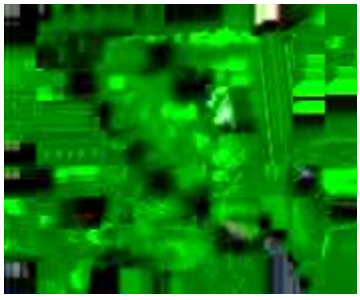
(b) Encryped 1. frame

(c) Encrypted 5. frame

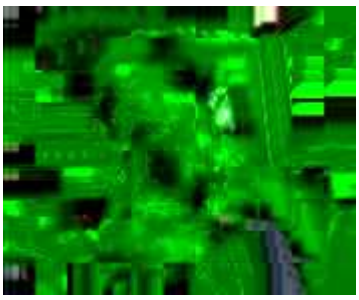


(d) Encrypted 9. frame

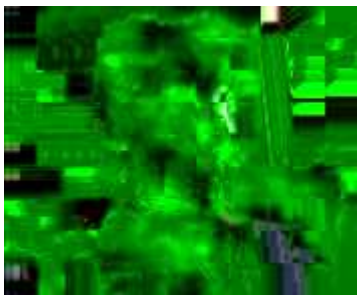
(e) Encrypted 13. frame



(f) Encrypted 17. frame



(g) Encrypted 21. frame



(h) Encrypted 25. frame

Figure A-5: Original and encrypted frames of Carphone.