MULTILAYER REVERSIBLE WATERMARKING USING NON-UNDERFLOW DIFFERENCE EXPANSION

MOHD SHAHIDAN BIN ABDULLAH

UNIVERSITI TEKNOLOGI MALAYSIA

MULTILAYER REVERSIBLE WATERMARKING USING NON-UNDERFLOW DIFFERENCE EXPANSION

MOHD SHAHIDAN BIN ABDULLAH

A thesis submitted in fulfilment of the
requirements for the award of the degree of
Doctor of Philosophy (Computer Science)

Faculty of Computer Science and Information Systems
Universiti Teknologi Malaysia

APRIL 2012

ACKNOWLEDGEMENT

In preparing this thesis, I was in contact with many people, researchers, academicians, and practitioners. They have contributed towards my understanding and thoughts. In particular, I wish to express my sincere appreciation to my main thesis supervisor, Professor Dr. Azizah binti bt Abdul Manaf, for encouragement, guidance, critics and friendship.

I am also indebted to Universiti Teknologi Malaysia (UTM) for funding my Ph.D. study. Librarians at UTM, my fellow postgraduate students should also be recognized for their support. My sincere appreciation also extends to all my colleagues and others who have provided assistance at various occasions. Their views and tips are useful indeed. Unfortunately, it is not possible to list all of them in this limited space.

I am grateful to all my family members especially my deepest feelings to my beloved wife Puan Roslina, sons and daughters for giving full support, aspiration and understanding during the period of my studies.

ABSTRACT

Use of digital reversible watermarking has grown tremendously in the last few years. The watermarking embeds an invisible payload into digital content for the purpose of copyright protection, content authentication, forensic tracking and other security purposes. Besides these applications, issues concerning imperceptible distortions, underflow problems and increasing capacity while maintaining the quality, are affecting the watermarking. The purpose of this research is to determine the optimum block size for embedding this reversible watermark, and to develop a novel multilayer reversible watermarking using non underflow difference expansion scheme. A traditional Difference Expansion (DE) scheme has been applied and used as a benchmark for comparison purposes with the proposed scheme. The new scheme comprising embedded multilayer reversible watermarking is capable of extracting and returning a watermarked image to its original form. Before the embedding phase, the host image is partitioned into non overlapping blocks and then the process of scanning is carried out to identify embeddable blocks. Then, location map is used as a key to store embeddable blocks, size of watermarks and number of layers that were applied. This process uses two ways of embedding; horizontal and vertical embedding. The number of layers embedded into the host image is decided according to the threshold value of Peak Signal to Noise Ratio (PSNR). The effect of multilayer and difference block sizes used on the performance of the developed scheme was studied. One of the findings was the discovery of the effectiveness of non underflow on capacity and quality of watermarked image. The results of the research had the highest embedding rate at 1.2 bit per pixel (bpp) with the highest visual quality of PSNR 30.8 decibel and equivalent to 6.2 percent improvement when compared to other available current schemes. The study proved that smaller block size yields higher capacity while retaining a low distortion as well as restores watermarked image to its original form.

ABSTRAK

Penggunaan tera air digital berbalik telah berkembang maju dari segi penyelidikan sejak akhir-akhir ini. Tera air membenamkan maklumat rahsia ke dalam media digital untuk kegunaan aplikasi kawalan hak cipta, pengesahan, pengesanan jenayah dan lain-lain kegunaan keselamatan. Di samping aplikasi-aplikasi ini isu-isu berkenaan perubahan nilai kualiti imej, masalah limpahan ke bawah dan peningkatan kapasiti bersama mengekalkan kualiti memberi kesan kepada tanda air. Kajian ini dilakukan bagi menentukan saiz blok yang optimum untuk membenamkan tera air, dan membangunkan satu skim asli multi-aras tera air berbalik melalui skim pembesaran perbezaan tanpa limpahan ke bawah. Skim asas pembesaran perbezaan digunakan sebagai penanda aras untuk dibuat perbandingan kepada skim yang dicadangkan. Skim baru terdiri dari pembenaman maklumat tera air berbalik secara multiaras, memperolehi semula maklumat dan memperolehi semula imej asal. Sebelum fasa pembenaman dilaksanakan, imej asal akan dipecahkan kepada blok-blok yang tidak bertindih dan seterusnya proses saringan akan dilaksanakan bagi menentukan blok-blok yang boleh digunakan. Selanjutnya, peta kedudukan digunakan sebagai kunci bagi menyimpan blok-blok yang boleh dimasukkan maklumat, saiz tera air dan bilangan lapisan dilaksanakan kepada imej. Proses ini membenamkan maklumat melalui dua cara iaitu membenam secara mendatar dan secara menegak. Bilangan lapisan maklumat boleh dimasukkan kepada imej tertakluk kepada nilai puncak PSNR. Kesan kepada multiaras dan perbezaan saiz blok digunakan untuk mengukur prestasi skim yang dibangunkan, dikaji. Satu daripada penemuan dijumpai iaitu kekesanan tanpa limpahan ke bawah kepada kapasiti dan kualiti imej tera air. Hasil penyelidikan telah menyumbang kepada kadar pembenaman yang tinggi pada kadar 1.2 bit setiap piksel (bpp) dengan kualiti tertinggi PSNR 30.8 decibel yang bersamaan 6.2 peratus peningkatan berbanding dengan skim-skim semasa yang lain. Kajian telah membuktikan bahawa saiz blok yang kecil meningkatkan kapasiti dan pada yang sama mengekalkan paparan kualiti dan mampu mengembalikan imej tera air kembali kepada bentuk asal.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

MRWNDE	-	Multilayer Reversible Watermarking using Non-underflow Difference Expansion
HVS	-	Human Visual System
MSE	-	Mean Squared Error
SNR	-	Signal to Noise Ratio
PSNR	-	Peak Signal to Noise Ratio
SSIM	-	Structural Similarity
dB	-	decibel
VW2D	-	Variable Watermark two-Dimensional
BER	-	Bit Error Ratio
BCR	-	Bit Correct Ratio
NCC	-	Normalized Cross Correlation
DE	-	Difference Expansion
JPEG	-	Joint Photographic Experts Group
LSB	-	Least Significant Bits
ISB	-	Intermediate Significant Bits
PVD	-	Pixel Value Differencing
DCT	-	Discrete Cosine Transform
DWT	-	Discrete Wavelet Transform
JND	-	just noticeable difference
OPA	-	optimal pixel adjustment
OPAP	-	optimal pixel adjustment process
MSB	-	most significant bit
LPAP	-	local pixel adjustment process
MER	-	minimum error replacement
RGB	-	Red Green Blue
DES	-	Data Encryption Standard
QoS	-	Quality of Service
DICOM	-	Digital Imaging and Communications in Medicine

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Overview

The world is facing a daily rapid technology change due to swift advancements of computers, networks and communications. These advancements have opened networks and individual machines to wide range of offenders' abuses. Offenders abuse the technologies in many criminal activities. In other words, they use technologies in various Cyber crimes (Sekgwathe *et al.*, 2011). A study found that there are relatively few criminal and civil cases that have not digital facilitation. It has estimated of over 85 percent of criminal and civil prosecution cases are committed through the use of digital technology (Zebbiche, 2009). The impact of the Internet is perhaps the visible in the use of digital media such as audio, image, and video. It has not so much affected the creation of these media, but to a much larger extent their reproduction and distribution. While in open networks most of the information is kept electronically, the demand for confidentiality and privacy gains more and more in importance (Lestriandoko *et al.,* 2010).

Open networks as a whole does not use secure links; therefore digital information may be vulnerable to intercept as well. Accordingly, the security of information has become an important issue, there is indeed a need to protect information from passing before curious eyes or, more importantly, from falling into wrong hands. Thus, multimedia security is much to consider in distributing digital information safety (Tsai, 2010; Khan and Muhammad, 2011).

Digital media have numerous advantages over analog media, such as higher quality, easy editing, lossless copying, and fast and efficient distribution. At the same time, such advantages may turn into disadvantages when the underlying technologies are being exploited by non authorized users. However digital media also offer the technique which is called watermarking to embed additional data into the original media data in a way that is perceptually, and sometimes also statistically, undetectable. This data embedding potential can be exploited to build protection mechanisms against the threats mentioned before, or to provide additional functionalities (Chen *et al.,* 2010; Chrysochos *et al.,* 2009).

Huang *et al.* (2009) describes that data hiding has been an interesting research topic since the early 1990's. Among the hiding schemes, reversible data hiding has attracted more and more attention in both researches and applications. With reversible data hiding, at the data extraction stage, both the original content and the hidden message should be perfectly extracted, hence, how to design such schemes seem an interesting task. It can be classified into two branches, one is histogram-based scheme, and the other is performed by adjusting the difference between adjacent pixels. The major insufficiency for the two schemes above is the limited amount of capacity.

Digital watermarking methods from the application point of review can be divided into two categories: robust watermarking and fragile watermarking (Al-Nu'aimi, 2009). Robust watermarking is mainly aimed at copyright protection. Here, "robust" means the embedded watermark should be very resistant to various signal processing operations. On the other hand, fragile watermarking is aimed at content authentication. A fragile watermark will be altered or destroyed when the digital content is modified (Xiao and Shih, 2010). As a special subset of fragile watermarking, reversible watermarking as described by Zhao *et al.* (2010) and Feng *et al*. (2011) has drawn lots of attention recently. Reversible watermark, (which is also called lossless watermark, invertible watermark, erasable watermark), has an additional advantage such that when watermarked content has been detected to be authentic, one can remove the watermark to retrieve the original, non-watermarked content. Such reversibility to retrieve non-watermarked content is highly desired in sensitive imagery, such as military data and medical data (Yang *et al.,* 2009).

Medical images are very important part of patient's records and information, which are stored in the databases of hospitals and may be exchanged between hospitals and health centers. Among these data, the patient information and medical images need to be properly organized, so to avoid mishandling and loss of data (Basheera *et al*., 2011). In order to achieve an efficient utilization of bandwidth of the communication channel and storage, data size must be reduced. Separate transmission of the image and data requires more bandwidth in transmission and more memory space during storage. By means of watermarking, interleaving one form data such as 1-D signal or text file over digital images can combine the advantages of data and security with efficient memory utilization (Zhang *et al*., 2011). Thus, the hiding capacity must be large enough to accommodate the payload. On the other hand, reversibility is one of the most important requirements for medical images, as they must be kept intact to avoid any misdiagnoses.

Several reversible watermarking schemes have been proposed by Gu and Gao (2008), Al-Qershi and Khoo (2010), Chrysochos *et al.* (2009), Zhao *et al*. (2010), Feng *et al*. (2011), Gupta *et al*. (2010), Lee *et al.* (2007), Zeki *et al*. (2011) and Peng *et al.* (2011). The concept of a reversible watermark was first introduced by Mintzer *et al.* (1997). The watermark that they embedded into an image was completely visible but could be removed since it was embedded in a reversible manner. Tian (2003) applied integer Haar wavelet transform into an image and embedded the watermark into high-frequency coefficients by difference expansion.

Kamstra *et al.* (2005) improved Fridrich *et al.* (2005) and Tian (2003) methods by sorting least-significant bits (LSB) or pairs of pixels to be watermarked with respect to the heuristically obtained values. The sorting locally improves the coding efficiency of the lossless compression, so that the overall performance is improved. Celik (2003) generalized a well-known LSB-substitution technique and achieved high capacity by using a prediction based conditional entropy coder. Yang *et al.* (2004) proposed a reversible watermarking scheme based on integer DCT transform. Peng *et al.* (2011) derived a theoretical bound on the embedding capacity for reversible data hiding. Yeo *et al.* (2011) applied reversible watermarking scheme with high embedding capacity for digital images. The input image is divided into non-overlapping blocks, and the watermark is embedded into the high-frequency wavelet coefficients of each block. To achieve the reversibility, invertible integer-to-integer wavelet transforms are used, and the conditions to avoid underflow and overflow in the spatial domain are derived for arbitrary wavelets and block sizes. The watermark payload includes not only messages but also side information required to reconstruct the original image at the decoder. The block-based embedding makes the size of the side information that needs to be embedded small in proportion to the total embedding capacity. Most of the algorithms used the block size equal to four by four.

Lee *et al.* (2007) applied semi-blind detection when he introduced the side information in his watermarked image. This side information was embedded together with watermark bit. Before he embedded the watermark bit, a process of selecting changeable blocks was implemented and recorded in the side information. These processes decreased the performance of the embedding process and increased the size of side information stored in watermarked image. Chrysochos *et al.* (2009) proposed a new Difference Expansion (DE) based scheme is presented that uses consecutive, overlapping pairs, instead of the non-overlapping pairs or triads used by traditional DE derivatives.

In this research a new scheme of reversible watermarking based on pixel value expansion is introduced by using multilayer embedding. The process of determining changeable blocks is modified using location map and it is determined before embedding process takes place. The proposed scheme also exploits sorting differences of expansion to improve quality for a given embedding capacity.

This research will present the multilayer reversible image watermarking using non-underflow difference expansion scheme. This chapter outlines the importance of reversible digital image watermarking and also explains how it can be done. The objectives and the thesis organization are also provided in this chapter.

1.2 Background of the Problem

Traditional watermarking and reversible watermarking scheme has similar characteristics in the process of embedding or extracting the additional information into or from the host image or original content (Gupta and Pieprzyk, 2009). Watermarking is a process of embedding extra information called payload into host image. After this process there should be no perceptible differences between original images and the watermarked images. In traditional watermarking scheme, these embedded payloads cannot be removed from the host images. However there are situations such as in medical and military applications that these watermarks need to be removed from the host and then reconstruct the watermarked back to original like before the process of watermarking take place (Guo and Zhuang, 2009; Arsalan *et al.*, 2010).

Watermarking is a process of embedding extra information called payload into the host mage. After this process there should be no perceptible differences between original images and the watermarked images. In traditional watermarking scheme, these embedded payloads cannot be removed from the host images. However, in some applications, especially in the medical, military, and legal domains, even the imperceptible distortion introduced in the watermarking process is unacceptable (Nayak *et al*., 2008). This has led to an interest in reversible watermarking, where the embedding is done in such a way that the information content of the host is preserved.

The distortion introduced by embedding the watermark is often constrained so that the host and the watermarked work are perceptually equivalent. The content authentication of multimedia data such as images is becoming more and more important in the fields of law enforcement, medical imaging, astrophysics research, and military application (Allatar, 2004b). Having the original image during analysis and diagnosis to make the right decision is of critical importance. Traditional watermarking techniques cannot provide adequate security and integrity for the content authentication because of their irreversible nature. Reversible watermarking enables exact recovery of the original image by extracting the embedded information from the watermarked images (Shaowei *et al.,* 2007; Zhao *et al.,* 2010).

Thodi and Rodriguez (2007) describes that reversible watermarking enables the embedding of useful information in a host signal without any loss of host information. Tian (2003) difference-expansion technique is a high-capacity, reversible method for data embedding. However, the method suffers from undesirable distortion at low embedding capacities and lack of capacity control due to the need for a location map. Problem of overflow and underflow occurs, and must be determined before embedding is executed.

Hu *et al*. (2009) mentioned that for difference-expansion (DE)-based reversible data hiding, the embedded bit-stream mainly consists of two parts: one part that conveys the secret message and the other part that contains embedding information, including position of pixels and location map. The first part is the payload while the second part is the auxiliary information package for blind detection. Tian’s classical DE method has a large auxiliary information package. They mitigated the problem by using a payload-independent overflow location map. However, the location map only can be used only for one layer of embedding.

Research by Khodaei *et al.* (2010) proposed a reversible data hiding method with low time complexity and high embedding capacity for gray-scale images. This method presents a block-based lossless data hiding schema to utilize the similarity between neighborhood pixels in the block to improve the marked-image quality. The experimental results show that our method has increased the hiding capacity with keeping acceptable marked image quality. The optimum block size is not determined during their experiments to obtain which appropriate block size gives better capacity and quality.

Tudoroiu *et al.* (2011) investigates the block implementation of median edge detector (MED) based prediction error expansion reversible watermarking. Image is divided in overlapping blocks. A block is considered for data embedding only if all its pixels can be embedded. By embedding on blocks, the size of the location map is reduced.

Internet is used to promote medical information. However, it is exposes to the unauthorized users to copy or misused the digitized information on the internet. Techniques of reversible watermarking are introduced to protect medical information. Lou *et al.* (2009) proposes a multiple-layer data hiding technique using difference expansion. Both pixels are changed during embedding process. Problem of their techniques still similar to Tian's Classical DE where overflow and underflow situation is occurs during the embedding process.

The main advantage of reversible image watermarking is the ability to recover the host image without any loses. Many reversible watermarking schemes were proposed for digital images during the last few years. The classic scheme for difference expansion (DE) was introduced by Tian (2003). Work done by Al-Qershi and Khoo (2010) adopt a two-layer difference expansion technique (2L-DE) in order to increase the hiding capacity by minimizing the size of embedding map while keeping a good visual quality of the watermarked image. They need to maintain two location maps to keeps the information for reversible process.

However, according to Fallahpour *et al.* (2009), in some applications, especially in the medical, military, and legal domains, even the imperceptible distortion introduced in the watermarking process is unacceptable. This has led to an interest in reversible watermarking, where the embedding is done in such a way that the information content of the host is preserved. In the nature of images in medical and military, there are many smooth areas in the images. Most of the techniques prefer to embed payloads at edge areas. This is because of to increase imperceptibility and robustness (Chiang *et al.,* 2009). Hence, to increase capacity, the smooth areas should be utilized for embedding the payload.

The aim of lossless watermarking is to develop a method that minimizes visible distortions while keeping the embedding capacity as high as possible. The method essentially should have the ability to recover the original image from the watermarked image if so required. From existing works, most of the techniques face a common problem that is, if the embedding capacity is increased, distortions become large and visible (Hu *et al.,* 2009). From the literatures, beginning with classical difference expansion scheme from Tian's and improves until now, most of the scheme facing the problems of overflow and underflow. This situation arise because of selected pair of pixels are changed by increasing and decreasing their values. When the pixel value is added, it could lead to overflow when the final value is more than 255 and underflow situation once the final value changes below zero. Both selected pixels are increase or decrease and this will affect the quality of watermarked image (Wang et al., 2010). If we can find ways only one of the pixel changes and either the situation of overflow or underflow can be avoided, hence this will increase the quality of host image.

1.3 Problem Statements

Although several researches have been done on reversible watermarking using difference expansion scheme, there are still several issues are highlighted. Classic Difference Expansion scheme was present by Tian (2003) become a benchmark for many researchers to evaluate their works (Wang *et al*., 2010). The main issue being focused is regarding tradeoff between three major components which are interrelated; embedding capacity, imperceptibility and robustness (Huang *et al.,* 2009; Khodaei *et al.,* 2010). In reversible watermarking many researchers try to improve embedding rate of payload and the same time obtaining acceptable visual quality of watermarked image. Second issue is about situation of overflow and underflow during the embedding of payload. Underflow is a condition where the new pixel value after embedding becomes negative. Non-Underflow is a case whereby underflow situation does not occur. Most of works from previous studies applied formula based on Tian's method. In this formula both selected pixels are transformed and situation of overflow and underflow may occur, when the value of pixel decrease or increase beyond zero and 255 respectively. The third issue is, determining the optimum block size and correlation between embedding rate and visual quality. Not many researchers discuss regarding the optimum of block size to be used during partitioning the image into non-overlapping block and the effect to the embedding rate and visual quality when different block size is applied (Tudoroiu *et al*., 2011).

The limitations of reversible image watermarking are illustrated in the magic triangle, shown in Figure 1.1 (Zhang *et al.,* 2010). Imperceptibility, robustness and capacity are at the corners of this diagram. This figure describes the required trade-off between capacity and robustness, at the same time keeping the quality at the acceptable level. Therefore, if the capacity is the aim to be achieved, its quality will be reduced and vice versa. Although there were some researchers who had ability to improve embedding rate, there are still opportunities to obtain higher capacity at acceptable visual quality (Huang *et al.,* 2009).

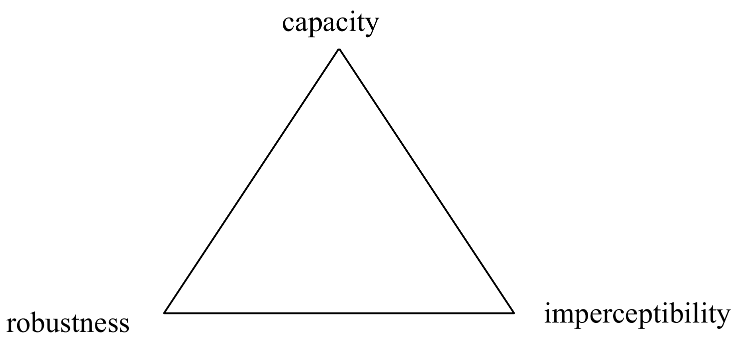


Figure 1.1 Required Tradeoff between Capacity, Robustness and Imperceptibility (Zhang *et al.,* 2010)

1.4 Research Questions

(i) What are the trade-off between capacity and visual quality of watermarked image related to the block size of image?

(ii) How to formulate a new non-underflow difference expansion formula in multilayer embedding and to obtain perfect recovery of watermarked image?

(iii) How to improve the rate of embedding without degrading the visual quality of the watermarked image?

1.5 Research Objectives

The following objectives were attempted to be achieved in this research;

(i) To implement, analyze and validate relationships on capacity and quality related to the block size of the host image.

(ii) To formulate a new non-underflow formula for multilayer embedding for a new scheme of reversible image watermarking.

(iii) To design a new scheme for reversible image watermarking by using Multilayer Reversible Watermarking using Non-underflow Difference Expansion (MRWNDE).

1.6 Research Scope

Below are the scopes of reversible image watermarking project:

(i) Standard testing data set and camera snapshot of images will be used to test the scheme for imperceptibility, capacity, extracting and reconstructing the original images.

(ii) To test the performance of reversible watermarked image with various five types of block size.

(iii) Digital reversible watermarking was developed to extract watermarked without the presence of the original image (blind watermark scheme)

(iv) The host images with size 512x512 will be in grayscale format.

(v) Proposed scheme is implemented using MATLAB program to illustrate the main idea involved in reversible watermarking scheme.

1.7 Significance of the Study

The outcome of this study would greatly contribute to reversible image watermarking technology with the following contributions:

(i) Developing and demonstrating a novel reversible image watermark using non-underflow difference expansion scheme through multiple embedding of watermark which can accommodate large amount of payload while keeping distortion low. The scheme restores the complete original image at the receiving end.

(ii) A new formula was developed to implement the proposed scheme. In conventional schemes, both pixels are transformed. In proposed scheme, new formula is derived, transforms only a pixel, instead of pair. These equations are steps to implement non-underflow reversible watermarking.

(iii) Optimizing block size will improve the process of searching the embeddable block, hence improve embedding rate at acceptable visual quality. Allows the smooth area as the best region to embed the watermarks. This is because in the smooth area the distortion are very minimal because new pixel values after embedding are almost unchanged, hence increase the imperceptibility.

1.8 Thesis Organization

This thesis is divided into 6 chapters and organized as follows:

Chapter 1: This chapter introduces the problem area and objective to be achieved. In this chapter reversible watermarking is introduced as the scheme to be used in the research.

Chapter 2: Literature review, explains the basics of watermarking, various types of reversible watermarking, applications and characteristics of watermarking. It also explains the advantages of reversible watermarking, the difference expansion scheme and the application using it.

Chapter 3: Describes research methodology, design and procedures.

Chapter 4: Implementation of proposed scheme, determining block size, embedding, extracting and reconstructing of host image.

Chapter 5: Analyzing and discussing on the results.

Chapter 6: Conclusion, novelty of proposed scheme, contribution and future work.

REFERENCES

Alattar, A. M. (2004a). Reversible watermark using difference expansion of quads. *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*. vol. 3, pp. 377-380.

Alattar, A. M. (2004b). Reversible watermark using the difference expansion of a generalized integer transform. *IEEE Transaction on Image Processing.* p 1147-1156.

Al-Nu'aimi, (2009). *Design, implementation and performance evaluation of robust and secure watermarking techniques for digital colored images*. Doctor Philosophy. University of Bradford.

Al-Qershi, O.M. and Khoo, B.E. (2010). Reversible watermarking scheme based on two-dimensional difference expansion (2D-DE). *International Conference on Computer Research and Development, ICCRD 2010*. art. no. 5489521, pp. 228-232.

Al-Qershi, O.M. and Khoo, B. E. (2011). Authentication and data hiding using a hybrid ROI-based watermarking scheme for DICOM images. *Journal of Digital Imaging.* v 24, n 1, p 114-125.

Arabzadeh, M., Abadi, M., Danyali, H. and Helfroush, M. S. (2010). Reversible watermarking based on interpolation error histogram shifting. *5th International Symposium on Telecommunications IST 2010.* p 840-845.

Arsalan, M., Malik, S., and Ambreen K. A. (2010). Intelligent threshold selection for reversible watermarking of medical images. *Proceedings of the 12th Annual Genetic and Evolutionary Computation Conference, GECCO '10 - Companion Publication.* p 1909-1914.

Awrangjeb M. (2003). An Overview of Reversible Data Hiding. *The 6th International Conference on Computer and Information Technology (ICCIT-03).*

Basheera, S., Prakash, D. B.; and Naganjaneyulu, P.V. (2011). Blind medical image watermarking technique for secure recovery of hidden data. *Communications in Computer and Information Science.* v 205 CCIS, p 185-192.

Basu, A. D., Tirtha, S., Maiti, S. I., Nurul, S., and Subir, K. (2009). FPGA based implementation of robust spatial domain image watermarking algorithm. *4th International Conference on Computers and Devices for Communication.* 2009, *Codec – 2009.*

Bennour, J., Dugelay, J. L., and Matta F. (2007). Watermarking attack (BOWS contest). *Proceedings of SPIE - The International Society for Optical Engineering*, v 6505.

Boucherkha, S., and Benmohamed, M. (2004). A Lossless Watermarking Based Authentication System For Medical Images. *Transations on Enginering, Computing and Technology, 1, 240-243.*

Bounkong, S. (2004). *Digital image watermarking*. Doctor Philosophy, Aston University.

Celik, M. U., Sharma, G., Tekalp, A. M., and Saber (2003). Localized lossless authentication watermark (LAW). *Proc. SPIE, Security and Watermarking of Multimedia Contents*. vol. 5020, no. 70.

Chang, Z., Kou, W., and Xu J. (2007). More compressible location map for reversible watermarking using expansion embedding. *Electronics Letters.* 43 (24), pp. 1353-1354.

Chang, Z., XU, J., and Kou W. (2008). Reversible Watermarking Schemes Using Spatial Quad-Based Difference Expansion. *Proceedings of International Conference on This paper appears in: Convergence and Hybrid Information Technology (ICHIT '08).*

Chen, X., Li, X., Yang, B., and Tang, Y. (2010). Reversible image watermarking based on a generalized integer transform. *IEEE International Conference on Acoustics, Speech and Signal Processing Proceedings.* art. no. 5496175, pp. 2382-2385.

Chen, D., Luo X. W., and Yu, M. (2006). Steganography preserving the property of the histogram for JPEG images. *Journal of Electronics and Information Technology*, v 28, n 2, p 252-256.

Chiang, K.H., Chang-Chien, K.C., Chang, R.F., and Yen, H.Y. (2009). Tamper Detection and Restoring System for Medical Images Using Wavelet-based Reversible Data Embedding. *Journal of Digital Imaging, 21, 77-90.*

Chrysochos, E., Varsaki, E.E., Fotopoulos, V., and Skodras A.N. (2009). High capacity reversible data hiding using overlapping difference expansion. *International Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS 2009*,, art. no. 5031447, pp. 121-124.

Coatrieux, G., and Lecornu, L. (2006). A Review of Image Watermarking Applications in Healthcare. *Proceedings of the 28th Annual International Conference of the IEEE : Engineering in Medicine and Biology Society, EMBS'06.*

Coltuc D. and Chassery J M, (2006). Simple reversible watermarking schemes: further results. *Proc. SPIE, Security, Steganography, and Watermarking of Multimedia Contents*, San Jose, CA, USA, pp. 739-746, Jan. 2006.

Coltuc, D. (2011). Improved embedding for prediction-based reversible watermarking. *IEEE Transactions on Information Forensics and Security.* v 6, n 3 PART 2, p 873-882, June 2011.

Cox, I., Miller, M. L., and Bloom, J. A. (2002). Digital Watermarking. *Morgan Kaufmann Publishers, San Francisco.* pp 12-36.

Cummins, J., Diskin, P., Samuel, L., and Robert P. (2004). Steganography And Digital Watermarking, School of Computer Science. *The University of Birmingham*. Copyright, 2004.

Dai, H. K., and Yeh, C. T. (2007). Content-Based Image Watermarking Via Public-Key Cryptosystem. *Computational Science and Its Application – ICCSA 2007*.

Dev-Chyuan, L., Hu, M. C., and Liu, J.L. (2009). Multiple layer data hiding scheme for medical images. *Computer Standards and Interfaces*. 31 (2), pp. 329-335.

Ding, S., Li, C., and Liu Z. (2010). Protecting hidden transmission of biometrics using authentication watermarking . *Proceedings - 2010 WASE International Conference on Information Engineering.* no. 5571271 , pp. 105-108.

Dugelay, J. L. and Roche, S. (1999). Fractal Transform Based Large Digital Watermark Embedding and Robust Full Blind Extraction. *Proceedings of IEEE International Conference on Multimedia Computing and Systems*.

El-Iskandarani, M.A. , Darwish, S.M., and Abubahia, A.M. (2009). Capacity and quality improvement in blind second generation watermarking. *Proceedings - International Carnahan Conference on Security Technology.* p 139-143.

Fallahpour, M., Megias, D., and Ghanbari M. (2009). High capacity, reversible data hiding in medical images. *Proceedings - International Conference on Image Processing. ICIP*,, art. no. 5413711, pp. 4241-4244.

Feng, Gui, Wu, and Guo, Z. (2011). Motion vector and mode selection based fragile video watermarking algorithm. *Proceedings: 2011 IEEE International Conference on Anti-Counterfeiting, Security and Identification.* p 73-76

Findik, Oguz, Babaoglu, Ismail; Ulker, and Erkan (2009). Watermarking schema using an artificial immune system in spatial domain . *ACM International Conference Proceeding Series.* v 403, p 945-950.

Fridrich, J., and Goljan, M. (2000). Robust Hash Functions for Digital Watermarking. *SUNY Binghamton*. NY. 2000

Gao, Fei, Liu, and Jing (2009). A blind watermark algorithm in color image based on mixed transform domain. *Beijing Ligong Daxue Xuebao/Transaction of Beijing Institute of Technology*. v 29, n 9, p 811-814+837.

Gao, T. G., and Gu, Q. L. (2007). Reversible watermarking algorithm based on wavelet lifting scheme. *Wavelet Analysis and Pattern Recognition, 2007. International Conference*. vol.4, no., pp.1771-1775.

Gonzalez, R.C. and Woods, R E. (2008) Digital Image Processing, *Prentice Hall.*

Gu, Q., and Gao, T. (2008). A new image authentication based on reversible watermarking algorithm. *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*,, art. no. 4593355, pp. 2727-2731.

Guo, X., and Zhuang, T.G. (2009). A region-based lossless watermarking scheme for enhancing security of medical data. *Journal of Digital Imaging*, 22 (1), pp. 53-64.

Guo, X. and Zhuang, T. G. (2008). A Region-Based Lossless Watermarking Scheme for Enhancing Security of Medical Data. *Journal of Digital Imaging, 0, 1-12.*

Gupta, G., and Pieprzyk, J. (2009). Reversible and blind database watermarking using difference expansion. *International Journal of Digital Crime and Forensics*, 1 (2), pp. 42-54.

Gupta, Pavan K., Shrivastava, and Shailendra, K. (2010). Improved RST-attacks resilient image watermarking based on joint SVD-DCT. *International Conference on Computer and Communication Technology, ICCCT-2010*, p 46-51.

Hartung, F. and Kutter, M. (1999). Multimedia Watermarking Techniques. *Proceedings of the IEEE*. vol. 87, no.7, pp. 1079-1107.

Ho, Y.A., Chan, Y.K., Wu, H.C., and Chu, Y.P. (2010). High-capacity reversible data hiding in binary images using pattern substitution. *Computer Standards and Interfaces*, 31 (4), pp. 787-794.

Hu, Y., and Jeon, B. (2006). Reversible visible watermarking technique for images. *Proceedings - International Conference on Image Processing, ICIP*,, art. no. 4107095, pp. 2577-2580.

Hu, Y., Lee, H. K., and Li, J. (2009). DE-based reversible data hiding with improved overflow location map. *IEEE Transactions on Circuits and Systems for Video Technology*,19(2),art.no.4703233,pp.250-260.

Huang, H. C., Wang, I. H., and Lu, Y.Y. (2009). High capacity reversible data hiding with adjacent-pixel-based difference expansion. *International Conference on Innovative Computing, Information and Control, ICICIC 2009*,, art. no. 5412394, pp. 639-642.

Huang, Hsiang, C., Chen, Y. H., and Lu, Y. Y. (2011). Histogram-based difference expansion for reversible data hiding with content statistics. *Proceedings - 7th International Conference on Intelligent Information Hiding and Multimedia Signal Processing. IIHMSP 2011*, p 37-40.

Izquierdo, E., Kim, H. J., and Macq, B. (2003). Introduction to the Special Issue on Authentication, Copyright Protection, and Information Hiding. *IEEE Transactions on Circuits and Systems for Video Technology*, 13 (8) , pp. 729-731.

Jin, H. L., Fujiyoshi, M., and Kiya, H. (2007). Lossless Data Hiding in the Spatial Domain for High Quality Images. *IEICE Transactions Fundamentals. E90-A, 4*, 771-777.

Kamstra, L., Henk J., and Heijmans A. M. (2005). Reversible data embedding into images using wavelet techniques and sorting. *IEEE Trans. Image Processing.* vol. 14, no. 12.

Kamstra, L. and Heijmans, A. M. (2005). Reversible Data Embedding into Images Using Wavelet Techniques and Sorting. *IEEE Transactions on Image Processing. 14, 12*, 2082-2090.

Khan, and Muhammad, K. (2011). Research advances in data hiding for multimedia security. *Multimedia Tools and Applications.* v 52, n 2-3, p 257-261.

Khelifi, and Fouad (2007). *Image compression and watermarking in the wavelet transform domain*. Doctor Philosophy. Queen's University Belfast.

Khodaei, Masoumeh, Faez, and Karim (2010). Reversible data hiding by using modified difference expansion.. *Proceedings of the 2010 2nd International Conference on Signal Processing Systems*, v 3, p V331-V334.

Kim, H. J., Sachnev, V., Shi, Y. Q., Nam, J. and Choo, H. G. (2009). A Novel Difference Expansion Transform for Reversible Data Embedding. *IEEE Transactions on Information Forensics and Security,* 3, 456 – 465

Kuribayashi, Minoru , Morii, Masakatu; Tanaka, and Hatsukazu (2005). *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v 3783 LNCS, p 441-453.

Kutter, M. and Hartung, F. (2000). Introduction to watermarking techniques. IN KATZENBEISSER, S. C. & PETITCOLAS, F. A. P. (Eds.) *Information Hiding Technologies fir Stegonography and Digital Watermarking.*

Kutter, M. and Petitcolasb, F. A. P. (1999). A fair benchmark for image watermarking systems. *Proceedings of SPIE : Security and Watermarking of Multimedia Contents*.

Lee, C. F., Chen, H. L., and Tso, H. K. (2010). Embedding capacity raising in reversible data hiding based on prediction of difference expansion. *Journal of Systems and Software*, 83 (10), pp. 1864-1872.

Lee, C. C., Wu, H. C., Tsai, C. S. and Chu, Y.P. (2008). Adaptive lossless steganographic scheme with centralized difference expansion. *Pattern Recognition*, 41,2097-2106.

Lee, Chin, F., Wang, K. H.; Chang, C. C.; and Huang, Y. L. (2009). A Reversible Data Hiding Scheme Based on Dual Steganographic Images. *Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication, ICUIMC'09*, p 228-237.

Lee, H. K., Kim, H. J., Kwon, K. R. and Lee, J. K. (2005). ROI Medical Image Watermarking Using DWT and Bit-plane. *Asia-Pacific Conference on Communications.* Perth, Western Australia.

Lee, S., Yoo, C. D., and Kalker, T. (2007). Reversible Image Watermarking Based on Integer-to-Integer Wavelet Transform. *Information Forensics and Security, IEEE Transactions.* vol.2, no.3, pp.321-330.

Lestriandoko, Nova H., Wirahman, and Taufiq (2010). Reversible watermarking using difference of virtual border for digital image protection. *International Conference on Distributed Frameworks for Multimedia Applications,* DFmA 2010.

Li, J., Li, S. H., and Sun, T. F. (2009). A semi-fragile watermark scheme based on the logistic chaos sequence and singular value decomposition. *Journal of Shanghai Jiaotong University.* v 43, n 7, p 1144-1148.

Li, Richard Y. M., Yip, S. K., and Chan, T. W. (2007). Enhanced image trans-coding using reversible data hiding. *Proceedings - IEEE International Symposium on Circuits and Systems.* p 1273-1276.

Lin, C. C., and Hsueh, N. L. (2008). A lossless data hiding scheme based on three-pixel block differences. *Pattern Recognition 41(4),* 1415–1425.

Lin, C. H., Chao, M. W., Liang, C. Y., and Lee, T.Y. (2010). A novel semi-blind-and-semi-reversible robust watermarking scheme for3D polygonal models. *Visual Computer.* p 1-11.

Logeshwaran, R. and Paul, I. J. L. (2010). Performance study on the suitability of Reed Solomon Codes in WiMAX. *International Conference on Wireless Communication and Sensor Computing. ICWCSC* 2010.

Lou D. C., Hu, M. C., and Liu J. L. (2009). Multiple layer data hiding scheme for medical images, *Computer Standards and Interfaces.* 31 (2), pp. 329-335.

Loukhaoukha, K., and Chouinard J. Y. (2010). Security of ownership watermarking of digital images based on singular value decomposition . *Journal of Electronic Imaging.* v19(1), n1

Lu T. C., and Chang C. C. (2008). Lossless nibbled data embedding scheme based on difference expansion. *Image and Vision Computing*, 26 (5), pp. 632-638.

Lu, T. C., and Huang, Y. H. (2009). An efficient block-based lossless information hiding technique. *Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication, ICUIMC'09*, p 342-347.

Luo, L., Chen, Z., Chen, M., Zeng, X., and Xiong, Z. (2010). Reversible image watermarking using interpolation technique. *IEEE Transactions on Information Forensics and Security*,5(1), art.no.5313862, pp.187-193.

Luo, Y. W., Li, S., and Liu, X. (2011). A lossless and visible watermarking algorithm. *Proceedings - 3rd International Conference on Measuring Technology and Mechatronics Automation, ICMTMA 2011.* v 1, p 95-98.

Ma, K., and Niu X. (2008). An improved reversible watermarking scheme. *International Conference on Signal Processing Proceedings, ICSP*,, art. no. 4697592, pp. 2229-2232.

Megalingam, Rajesh K. N., and Mithun, S. R. (2010). Performance comparison of novel, robust spatial domain digital image watermarking with the conventional frequency domain watermarking techniques. *International Conference on Signal Acquisition and Processing.* p 349-353.

Ming, C., Zhenyong, C., Xiao, Z., and Zhang, X. (2009). Reversible Data Hiding Using Additive Prediction-Error Expansion. *MMandSec'09 - Proceedings of the 11th ACM Multimedia Security Workshop*, p 19-24.

Mintzer, F., Braudway, G. W. and Yeung, M. M. (1997). Effective and Ineffective Digital Watermarks. *Proceedings of the International Conference on Image Processing*.

Mohammad K. Y., and Ahmad Al-Jaber. (2006). Reversible Watermarking Using Modified Difference Expansion. *International Journal of Computing & Information Sciences.* Vol 4, No. 3.

Moreno, Oscar, O. U., and José (2007). Double periodic arrays with good correlation for applications in watermarking. *3rd International Workshop on Signal Design and Its Applications in Communications*, p 214-218.

Morrison, J. (2006). *Mesh watermarking: a novel robust watermarking scheme based on the discrete cosine transform*. Doctor Philosophy, University of Limerick

Nisar, A. M. (2010). *Watermarking of Medical Images for Content Authentication and Copyright Protection*. Doctor Philosophy, Faculty of Computer Science and Engineering, Institute of Engineering Sciences and Technology, Pakhtunkhwa, Pakistan

Peng, F., Lei, Y., Zhou, L. M., and Sun, X. M. (2011). A reversible watermarking scheme for two-dimensional CAD engineering graphics based on improved difference expansion . *CAD Computer Aided Design.* v 43, n 8, p 1018-1024.

Pun, C. M., Hemman, N., and Yuan, X. C. (2010). Reversible image watermarking using integer transform on lifting wavelet coefficients. *Proceeding - 6th International Conference on Digital Content, Multimedia Technology and Its Applications, IDC2010*, p 390-394.

Sachnev V., Kim H.J., Nam J., Suresh S., and Shi Y.Q.(2009). Reversible watermarking algorithm using sorting and prediction. *IEEE Transactions on Circuits and Systems for Video Technology.* 19 (7), art. no. 4811982, pp. 989-999.

Salman, M. (2009). *Watermarking and steganography of 3-D models*. Doctor Philosophy, University of Surrey

Sekgwathe, Virginiah, Talib, and Mohammad (2011). Cyber crime detection and protection: Third world still to cope-up . *Communications in Computer and Information Science.* v 171 CCIS, p 171-181.

Shaowei, W., Yao, Z., Shyang, P. J., Ni, Rongrong (2007). A Novel Reversible Watermarking Based on an Integer Transform. *Image Processing, 2007. ICIP 2007. IEEE International Conference.* vol.3, no., pp.III -241-III -244.

Singh, Jyotsna, Garg, Parul, De, and Alok Nath (2009). Watermarking of unified multimedia data types, audio and image. *Proceedings of an IEEE India Council Conference*, 2009.

Sulaiman, Adeeb, H., Baji, and Faiq S. (2009). Fractal based fragile watermark. *International Conference on Computer and Electrical Engineering, ICCEE 2009.* v 2, p 139-143.

Tai, W. L., Yeh, C. M., and Chang, C. C. (2009). Reversible data hiding based on histogram modification of pixel differences . *IEEE Transactions on Circuits and Systems for Video Technology*, v 19, n 6, p 906-910.

Thodi, D. M. and Rodriguez, J. J. (2007). Expansion Embedding Techniques for Reversible Watermarking. *IEEE Transaction on Image Processing*,16,721-730.

Thompson, A. I. (2008). *Digital watermarking for multimedia security using complex wavelets*. Doctor Philosophy. Queens University Belfast.

Tian, (2003). Reversible Data Embedding Using a Difference Expansion. *IEEE Trans. Circuts and Systems for Video Technology.* vol. 13, no. 8, pp. 890-896.

Tsai, H. M., and Chang L.W. (2010). Secure reversible visible image watermarking with authentication. *Signal Processing: Image Communication.* 25 (1), pp. 10-17.

Tudoroiu, Adrian, Caciula, Ion; Coltuc, and Dinu (2011). Block map implementation of difference expansion reversible watermarking. *International Symposium on Signals, Circuits and Systems, Proceedings.* p 321-324.

Vasiliy, Sachnev, Kim, H. J., Xiang, S., and Nam, J. (2008). .An improved reversible difference expansion watermarking algorithm, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).* v 5041 LNCS, p 254-263.

Wang, C., Li, X., and Yang, B. (2010). High capacity reversible image watermarking based on integer transform. *International Conference on Image Processing. ICIP*, p 217-220.

Wang, Z., Bovik, A. C., Sheikh, H. R. and Simoncelli, E. P. (2004). Image Quality Assessment: From Error Visibility to Structual Similarity. *IEEE Transaction On Image Processing,* 13, 600-612.

Weng, S., Zhao, Y., Pan J. S., and Ni, R. (2008). Reversible data hiding using the companding technique and improved DE method. *Circuits, Systems, and Signal Processing*, 27 (2), pp. 229-245.

Weng, S., Zhao, Y., Pan, J. S. and Ni, R. (2007). A novel reversible watermarking based on an integer transform. *Proceedings of IEEE International Conferecne on image Processing* (ICIP 2007).

Xiao, D., and Shih, F.Y. (2010). A reversible image authentication scheme based on chaotic fragile watermark . *International Journal of Innovative Computing, Information and Control*. 6 (10) , pp. 4731-4742.

Yang B., M. Schmucker, and X. Niu. (2005). Approaching Optimal Value Expansion for Reversible Watermarking. *ACM*, August 2005.

Yang C. H., and Tsai M.H. (2010). Improving histogram-based reversible data hiding by interleaving predictions. *IET Image Processing.* 4 (4) , pp. 223-234.

Yang, Y., Sun, X., Yang, H., Li, C. T., and Xiao, R. (2009). A contrast-sensitive reversible visible image watermarking technique. *IEEE Transactions on Circuits and Systems for Video Technology*, 19 (5), art. no. 4801612, pp. 656-667.

Yang, C. K. and Huang, C. S. (2004). A Novel Watermarking Technique for Tampering Detection in Digital Image. *Electronic Letters on Computer Vision and Image Analysis,* Vol. 3, pp.1-12.

Yeo, D. G., and Lee, H. Y. (2011). Block-based image authentication algorithm using reversible watermarking. *Lecture Notes in Electrical Engineering.* v 114 LNEE, p 703-711.

Yeo, D. G., Lee, H. Y., and Kim, B. M. (2010). Differential histogram modification-based reversible watermarking with predicted error compensation. *Proceedings - 2010 6th International Conference on Intelligent Information Hiding and Multimedia Signal Processing, IIHMSP 2010*, p 106-109.

Zebbiche, K. (2009). *Data hiding for securing fingerprint data access*. Doctor Philosophy. Queen's University Belfast.

Zeki, A. M. , Azizah, A. M., and Mahmod, S. S. (2011). High watermarking capacity based on spatial domain technique. *Information Technology Journal*. v10, n7, p1367-1373.

Zeki, A. M. (2009). *Watermarking Techniques Using Intermediate Significant Bit.* Doctor Philosophy, Faculty of Computer Science and Information Systems, University Technology Malaysia.

Zeng, X., Ping, L., and Li, Z. (2010). A reversible data hiding scheme using center pixel difference. *Journal of Multimedia.* v 5, n 4, p 377-384.

Zhang, B., Xin, Y., Niu, X. X., Yuan, K. G., and Jiang, H. B. (2010). A near reversible image watermarking algorithm. *International Conference on Machine Learning and Cybernetics, ICMLC 2010.* v 6, p 2824-2828.

Zhang, X., Zhang, F., and Xu, Y. (2011). Quality evaluation of digital image watermarking. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).* v 6676 LNCS, n PART 2, p 241-250.

Zhao, H., Du, S., and Zhang, D. (2010). A reversible watermarking scheme for 2D vector drawings based on difference expansion.*, IEEE 11th International Conference on Computer-Aided Industrial Design and Conceptual Design, CAID and CD'2010.* v 2, p 1441-1446.

Zhao, Y., Campisi, P. and Kundur, D. (2004). Dual Domain Watermarking for Authentication and Compression of Cultural Heritage Images. *IEEE Transaction on Image Processing*, 13,430-448.

Zheng, D., Liu, Y., Zhao, J. and Saddik, A. E. (2007). A Survey of RST Invariant Image Watermarking Algorithms . *ACM Computing Survey*,39.

Zheng, L., Li, Y. F., and Liuping, L. H. (2010). Research and implementation of fragile watermark for vector graphics. *2010 International Conference on Computer Engineering and Technology, Proceedings.* v 1, p V1522-V1525.

APPENDIX A

SOURCE CODE

Program 1 : Control Program (Scanning and Embedding)

```
%embed Main
clear all;close all;clc;
i=1;
a=importdata('host_imageDB.txt');
c=importdata('wm_image.txt');
ftxt=fopen('fembed_b100.txt','w+');
fn1=importdata('owner.txt');
fn2=char(fn1(1:end,1:end));
b=char(a(1:end,1:end));
d=char(c(1:end,1:end));
block=2;
fprintf(ftxt,'%s %d\n','Embedding the Watermark into Host Image with blocksize:',block);
while i<=20
    totaltime=0;
    starttime=0;
    fprintf(ftxt,'\n%d. %s\n',i,b(i,1:end));
    fprintf(ftxt,'%s\t%s\t%s\t%s\t%s\t%s','Layer','Eltime','Payload','HostSize','BitRate');
    fprintf(ftxt,'%s\t%s\t%s\n','totalBitrate','PSNR','SSIM');
   fh=strcat('c:\users\shahidan\documents\matlab\host\database\',b(i,1:end));
    host=imread(fh);
    host=imresize(host,[512 512]);
    vimage=host;
    [a1,a2]=size(host);
    hostsize=a1*a2;
   % fn22=strcat('c:\users\shahidan\documents\matlab\textWM\',fn2(i,1:end))
     fn22=strcat('c:\users\shahidan\documents\matlab\textWM\',fn2(1,1:end))
    fid=fopen(fn22,'r');
    fn3 = fread(fid);
    disp(char(fn3'))
    [sf1 sf2]=size(fn3);
    fn4=double(fn3);
    wm1=dec2bin(fn4);
    [s1 s2]=size(wm1);
    s3=s1*s2;
    vf1=strcat('c:\users\shahidan\documents\matlab\WM\');
    wm2=imread(strcat(vf1,d(2,1:end)));
    wm3=imread(strcat(vf1,d(3,1:end)));
    wm4=imread(strcat(vf1,d(4,1:end)));
```

```
wm5=imread(strcat(vf1,d(5,1:end)));
wm6=imread(strcat(vf1,d(6,1:end)));
wm7=imread(strcat(vf1,d(7,1:end)));
wm8=imread(strcat(vf1,d(8,1:end)));
wm9=imread(strcat(vf1,d(9,1:end)));
wm10=imread(strcat(vf1,d(10,1:end)));
wm11=imread(strcat(vf1,d(11,1:end)));
wm12=imread(strcat(vf1,d(12,1:end)));
wm13=imread(strcat(vf1,d(13,1:end)));
wm14=imread(strcat(vf1,d(14,1:end)));
wm15=imread(strcat(vf1,d(15,1:end)));
wm16=imread(strcat(vf1,d(16,1:end)));
wm17=imread(strcat(vf1,d(17,1:end)));
wm18=imread(strcat(vf1,d(18,1:end)));
locimage=imread(strcat('c:\users\shahidan\documents\matlab\host\','Biglena.jpg'));
saizLoc1=(fix(hostsize/16)+ 35)* 20;
saizLoc2=fix(sqrt(saizLoc1))+1;
r=saizLoc2;
c=saizLoc2;
locimage=imresize(locimage,[r c]);
[r,c]=size(locimage);
locsize=r*c;
locimage(1:r,1:c)=0;
Vlayer=0;
Hlayer=0;
layer=0;
tbitrate=0;
psnr=30;
while psnr >=30
   if Vlayer == 0
      bv=35;
   end
   if Hlayer == 0
      bh=locsize;
   end
   if layer >= 2
      bv = 35 + Vlayer * fix(hostsize/(2*8)) + 1;
      bh = locsize - Hlayer * fix(hostsize/(2*8)) - 1;
      if bv > locsize
         break;
      end
      if bh < 35
         break;
      end
   end

   Vlayer=Vlayer+1;
   layer=layer+1;
   if layer > 16
    break;
   end
   switch layer
      case 1
         wm=wm1;
```

```
        case 2
            wm=wm2;
         case 3
            wm=wm3;
        case 4
            wm=wm4;
        case 5
            wm=wm5;
        case 6
            wm=wm6;
        case 7
            wm=wm7;
        case 8
            wm=wm8;
        case 9
            wm=wm9;
        case 10
            wm=wm10;
         case 11
            wm=wm11;
        case 12
            wm=wm12;
        case 13
            wm=wm13;
        case 14
            wm=wm14;
        case 15
            wm=wm15;
        case 16
            wm=wm15;
        case 14
            wm=wm14;
        case 15
            wm=wm15;
        case 16
            wm=wm15;
        case 17
            wm=wm17;
        case 18
            wm=wm18;
    end
    starttime=cputime;
    [locimage acp1 vpos payload] = scanV2(vimage,wm,locimage,bv,layer,block);
    if payload > acp1
      mesej = 'embeddable pairs in host image > payload size ';
      fprintf(ftxt,'\t%d\t%d\t%d\t%s\n',layer,acp1,payload,mesej);
      break;
    end
    [vimage locimage payload bitrate] =
embedVt2(vimage,wm,locimage,block,bv,Vlayer,layer);
    masa=cputime-starttime;
    totaltime=totaltime+masa;
    psnr = rev_psnr(host,vimage);
    [mssim, ssim_map] = ssim_index(host, vimage);
```

```
        tbitrate=tbitrate + bitrate;
        layer

fprintf(ftxt,'\t%d\t%.2f\t%d\t%d\t\t%.4f\t\t%.4f\t\t%.4f\t%.4f\n',layer,masa,payload,hostsiz
e,bitrate,tbitrate,psnr,mssim);

        if psnr < 30
           break;
        end
        Hlayer=Hlayer+1;
        layer=layer+1;
        switch layer
           case 1
              wm=wm1;
           case 2
              wm=wm2;
            case 3
              wm=wm3;
           case 4
              wm=wm4;
           case 5
              wm=wm5;
           case 6
              wm=wm6;
           case 7
              wm=wm7;
           case 8
              wm=wm8;
           case 9
              wm=wm9;
           case 10
              wm=wm10;
            case 11
              wm=wm11;
           case 12
              wm=wm12;
           case 13
              wm=wm13;
           case 14
              wm=wm14;
           case 15
              wm=wm15;
           case 16
              wm=wm15;
           case 14
              wm=wm14;
           case 15
              wm=wm15;
           case 16
              wm=wm15;
           case 17
              wm=wm17;
           case 18
              wm=wm18;
```

```
        end
        starttime=cputime;
        [locimage acp1 hpos payload] = scanH2(vimage,wm,locimage,bh,layer,block);

        if payload > acp1
          mesej = 'embeddable pairs in host image > payload size ';
          fprintf(ftxt,'\t%d\t%d\t%d\t%s\n',layer,acp1,payload,mesej);
         break;
        end
        [vimage locimage payload bitrate] =
embedHt2(vimage,wm,locimage,bh,Hlayer,layer,block);
        masa=cputime-starttime;
        totaltime=totaltime+masa;
        psnr = rev_psnr(host,vimage);
        [mssim, ssim_map] = ssim_index(host, vimage);
        tbitrate=tbitrate + bitrate;

fprintf(ftxt,'\t%d\t%.2f\t%d\t%d\t\t%.4f\t\t%.4f\t\t%.4f\t%.4f\n',layer,masa,payload,hostsiz
e,bitrate,tbitrate,psnr,mssim);
    end
    fprintf(ftxt,'\t%s%.2f\n','Total time ',totaltime);
    totaltime=0;
    s1=b(i,1:end);
    s2=findstr(s1,'.jpg');
    if s2 > 0
        s3=strcat(s1(1:s2-1),'.bmp');
        imwrite(locimage,strcat('c:\users\shahidan\documents\matlab\lokasi\','lokasi_',s3));
        imwrite(vimage,strcat('c:\users\shahidan\documents\matlab\hostWm\','wm_',s3));
    else

imwrite(locimage,strcat('c:\users\shahidan\documents\matlab\lokasi\','lokasi_',b(i,1:end)));

imwrite(vimage,strcat('c:\users\shahidan\documents\matlab\hostWm\','wm_',b(i,1:end)));
    end
    i=i+1;
    fclose(fid);
end
fclose(ftxt);
```

Program 2 : Vertical Scanning

```
function [w acp1 pos payload] = scanV2(image1,f2w,w,b3,layer,block)
round=0;
beza=0;
s=dec2bin(2);
[row,col]=size(image1);
hostsize=row*col;
wimage = image1;
%display 'size watermark image row and column';
[pan leb] = size(f2w);
if layer==1
```

```
  wmark1=f2w;
else
  wmark1=dec2bin(f2w);
end
[wk1 wk2]=size(wmark1);
wmark=wk1*wk2;
if b3==35
    payload=pan*leb;
else
    payload=pan*leb*8;
end
[baris,lajur] = size(w);
locsize=baris*lajur;
wm=dec2bin(w);
[w1,w2] = size(wm);
acp=0;
b1=b3;
while wmark > acp
    if wmark > hostsize/2
        acp=ceil(hostsize/2);
        break;
    end
round=round+1
if round > 50
    break;
end
acp
i=1;
j=1;
b1=b3;
neg=0;
w3=1;
kes0=0;
acp=0;
rej=0;
b2=0;
while i < (row-block) && (acp < wmark)
    while j< (col-block) && (acp < wmark)
     r=0;
     c=0;
     while c <= (block-1) && (acp < wmark)
         while r <= (block-1-(block/2)) && (acp < wmark)

         b2=b2+1;
         if b2>8
            b2=1;
            b1=b1+1;
         end

    x1=double(wimage(r+i,c+j));
    x2=double(wimage(r+(block/2)+i,c+j));

    if(x1>x2)
        y=x1;
```

```
        x=x2;
    else
        y=x2;
        x=x1;
    end
    d=y-x;
    ov = floor((x+254)/2);
    if (x1==255 && x2==255)
        wm(b1,b2)= s(2);
        rej=rej+1;
    else
     if ((y <= ov) && (d <= beza))
        wm(b1,b2)= s(1);
        acp=acp+1;

    else
        wm(b1,b2)= s(2);
        rej=rej+1;
    end
   end
    r=r+1;
   end;    % while r
    r=0;
    c=c+1;
     end;   % while c
j=j+block;

    end;    % while j
    j=1;
    i=i+block;

end;    % while i
beza=beza+1;
end   % while wmark
pos=b1;
loc=bin2dec(wm);
loc=reshape(loc,baris,lajur);
w(1:baris,1:lajur)=loc(1:baris,1:lajur);
acp1=acp;
end
```

Program 3 : Vertical Embedding

```
function [wimage image2 payload bitrate] =
embedVt2(image1,image3,image2,block,pos,Vlayer,layer)
[pp,ll]=size(image2);
[row,col]=size(image1);
hostsize=row*col;
wimage = image1;
loc=dec2bin(image2);
round=1;
w3=1;
```

```
rej=0;
boleh=0;
embed=0;
s=dec2bin(2);
 [pan leb] = size(image3);
if layer==1
  wm=image3;
else
  wm=dec2bin(image3);
end
[w1,w2] = size(wm);
wmsize=w1*w2;

 i=1;
 j=1;
 b1=pos;
 b2=0;
while i < (row-block)
  while j<(col-block)
     r=0;
     c=0;
   while c <= block-1
     while r <= block-1-(block/2)

        b2=b2+1;
        if b2>8
            b1=b1+1;
            b2=1;
        end
        if boleh > wmsize
            round=3;
            i=row;
            j=col;
            r=block-1-(block/2);
            c=block-1;
            break;
        end
    x1=double(wimage(r+i,c+j));
    x2=double(wimage(r+(block/2)+i,c+j));
    if(x1>x2)
        y=x1;
        x=x2;
        diff=y-x;
    else
         y=x2;
         x=x1;
        diff=y-x;
     end

     if loc(b1,b2)== s(2)
       r=r+1;
       continue;
     end
```

```
    if loc(b1,b2)== s(1)
     boleh=boleh+1;
     embed=1;

      if wm(w3)== s(1)
          msgbit=1;
      else
          msgbit=0;
      end
      d1=diff+msgbit;
      y1=y+d1;
      if y1>255
          display y1>255
          x1,x2
      end
      if x1>x2
       wimage(r+i,c+j)=y1;
      else
       wimage(r+(block/2)+i,c+j)=y1;
      end
    end
     r=r+1;
     if embed==1
       embed=0;
       w3=w3+1;        % goto next bit of wm
     end
       if(w3>wmsize)
        i=row;
        j=col;
        r=block-1-(block/2);
        c=block-1;
        break;
       end    % if

    end;    % while r
    r=0;
    c=c+1;
   end;   % while c
j=j+block;

 end;    % while j
    j=1;
    i=i+block;

end;    % while i

im2=bin2dec(loc);
im2=reshape(im2,pp,ll);
image2(1:pp,1:ll)=im2(1:pp,1:ll);
display 'di V';
d=layer*2;
if layer==1
    image2(1,1)=fix(pan/4);
    image2(2,1)=mod(pan,4);
```

```
   image2(18,1)=0;
else
  image2(d-1,1)=pan;
  image2(d,1) = leb;
end
image2(33,1)= Vlayer;
payload=wmsize;
bitrate=wmsize/hostsize;
end
```

Program 4 : Horizontal Scanning

```
%scanH
function [w acp1 pos payload] = scanH2(image1,f2w,w,bh,layer,block)
[row,col]=size(image1);
hostsize=row*col;
wimage = image1;
s=dec2bin(2);
[pan,leb] = size(f2w);
if layer==1
  f3w=f2w;
else
  f3w=dec2bin(f2w);
end

[f4 f5]=size(f3w);
wmark=f4*f5;
payload=pan*leb*8;
[baris,lajur] = size(w);
wm=dec2bin(w);
round=0;
beza=0;
acp=0;
b1=bh;
kes0=0;
```

```
while wmark > acp
    if payload > hostsize/2
        acp=ceil(hostsize/2);
        break;
    end
round=round+1;
if round > 50
    break;
end
i=row-block;
j=col-block;
neg=0;
w3=1;
rej=0;
kes0=0;
acp=0;
b1=bh;
b2=0;
while i > block && (acp < wmark)
    while j > block && (acp < wmark)
     r=0;
     c=0;
     while r <= block-1  && (acp < wmark)
         while c <= block-1-(block/2) && (acp < wmark)

         b2=b2+1;
         if b2>8
           b2=1;
           b1=b1-1;
           if b1 < 35
             i=block;
             j=block;
             r=block-1;
             c=block-1-(block/2);
             break;
           end    % for if
         end
    %fprintf('hello2 %d\n',j)

    x1=double(wimage(i-r,j-c));
    x2=double(wimage(i-r,j-c-(block/2)));

    if(x1>x2)
        y=x1;
        x=x2;
    else
        y=x2;
        x=x1;
    end
    d=y-x;
    ov = floor((x+254)/2);
    if (x1==255 && x2==255)
        wm(b1,b2)= s(2);
```

```
        rej=rej+1;
    else
    if y <= ov
        if d <= beza
         wm(b1,b2)= s(1);
         acp=acp+1;
        else
         wm(b1,b2)= s(2);
         rej=rej+1;
        end
    end
    end
    c=c+1;

         end;    % while c
    c=0;
    r=r+1;
     end;   % while r
j=j-block;

    end;    % while j
    j=col-block;
    i=i-block;

end;    % while i
beza=beza+1;
end  % while wmark
pos=b1;
loc=bin2dec(wm);
loc=reshape(loc,baris,lajur);
w(1:baris,1:lajur)=loc(1:baris,1:lajur);
acp1=acp;
end
```

Program 5 : Horizontal Embedding

```
function [wimage image2 payload bitrate] =
embedHt2(image1,w,image2,pos,Hlayer,layer,block)
[pp,ll]=size(image2);
[row,col]=size(image1);
hostsize=row*col;
wimage = image1;
loc=dec2bin(image2);
[cw1,cw2]=size(loc);
round=1;
msgbit=0;
neg=0;
w3=1;
rej=0;
boleh=0;
embed=0;
s=dec2bin(2);
 [pan leb] = size(w);
if layer==1
  wm=w;
else
  wm=dec2bin(w);
end
 [w1,w2] = size(wm);
wmsize=w1*w2;

 b1=pos;
 b2=0;
 i=row-block;
 j=col-block;
while i > block
  while j > block
     r=0;
     c=0;
   while r <= block-1
     while c <= block-1-(block/2)

        b2=b2+1;
        if b2>8
            b1=b1-1;
            b2=1;
        end
        if boleh > wmsize
            round=3;
            i= block;
            j= block;
            c=block-1-(block/2);
            r=block-1;
            break;
        end
    x1=double(wimage(i-r,j-c));
```

```
x2=double(wimage(i-r,j-c-(block/2)));

if(x1>x2)
    y=x1;
    x=x2;
    diff=y-x;
else
     y=x2;
     x=x1;
    diff=y-x;
 end

 if loc(b1,b2)== s(2)
   c=c+1;
   continue;
 end

if loc(b1,b2)== s(1)
 boleh=boleh+1;
 embed=1;
 if wm(w3)== s(1)
      msgbit=1;
  else
      msgbit=0;
  end
  d1=diff + msgbit;
  y1=y+d1;
  if y1>255
      display y1>255
      x1,x2
  end
  if x1>x2
   wimage(i-r,j-c)=y1;
  else
   wimage(i-r,j-c-(block/2))=y1;
  end
end

c=c+1;
if embed==1
   embed=0;
   w3=w3+1;   % goto next bit of wm
end
if(w3>wmsize)
    round=3;
    i=block;
    j=block;
    c=block-1-(block/2);
    r=block-1;
    break;
end    % if

end;    % while r
c=0;
```

```
   r=r+1;
  end;   % while c
j=j-block;

 end;    % while j
   j=col-block;
   i=i-block;
end;    % while i

im2=bin2dec(loc);
im2=reshape(im2,pp,ll);
image2(1:pp,1:ll)=im2(1:pp,1:ll);
display 'di H';
d=layer*2;
image2(d-1,1)=pan;
image2(d,1) = leb;
image2(34,1)= Hlayer;
payload=wmsize;
bitrate=wmsize/hostsize;
end
```

Program 6 : Control Proram (Extracting and Reconstructing)

```
% extractMain
clear all;close all;clc;
i=1;
% a=importdata('host_image.txt');
a=importdata('host_imageDB.txt');
%c=importdata('watermarked_image.txt');
ftxt=fopen('fextract_b100.txt','w+');
b=char(a(1:end,1:end));
% d=char(c(1:end,1:end));
fn1=importdata('owner.txt');
fn2=char(fn1(1:end,1:end));
block=2;
fprintf(ftxt,'%s %d\n\n','Extraction and Reversible with blocksize: ',block);
while i<=20
   totaltime=0;
   fprintf(ftxt,'%d. %s\n',i,b(i,1:end));

fprintf(ftxt,'\t%s\t%s\t%s\t\t%s\t\t%s\t%s\t%s\n','Layer','time','Payload','HostSize','bitrate','
PSNR','SSIM');
   fout=fopen(strcat('text_',b(i,1:end)),'w');
host=imread(strcat('c:\users\shahidan\documents\matlab\host\',b(i,1:end)));
   host=imread(strcat('c:\users\shahidan\documents\matlab\host\database\',b(i,1:end)));
   host=imresize(host,[512 512]);
   [r c]= size(host);
   hostsize=r*c;
   vimage=host;
   s1=b(i,1:end);
   s2=findstr(s1,'.jpg');
   if s2>0
      s3=strcat(s1(1:s2-1),'.bmp');
      s4=1;
   else
      s4=0;
   end
   if s4==1
      wrimage=imread(strcat('c:\users\shahidan\documents\matlab\hostWm\','wm_',s3));
      locimage=imread(strcat('c:\users\shahidan\documents\matlab\lokasi\','lokasi_',s3));
   else
       wrimage=imread(strcat('c:\users\shahidan\documents\matlab\hostWm\','wm_',s1));
       locimage=imread(strcat('c:\users\shahidan\documents\matlab\lokasi\','lokasi_',s1));
   end
```

```
[r,c]=size(locimage);
  locsize=r*c
  Vlayer=double(locimage(33,1));
  Hlayer=double(locimage(34,1));
  layer=Vlayer+Hlayer;
  tbitrate=0;
  bv=35;
  bh=35;
  while layer >= 1

      if Vlayer > 1
        bv = 35 + (Vlayer-1)* fix(hostsize/(2*8)) + 1

      end
      if Hlayer > 1
          bh = locsize - (Hlayer-1) * fix(hostsize/(2*8)) - 1
      end
      if Hlayer ==1
          bh=locsize
      end
      if Vlayer == 1
          bv=35
      end
      if bv > locsize
       display 'kes break bv>locsize'
       break;
      end
      if bh < 35
       display 'kes break bh<35'
       break;
      end
    starttime=cputime;
    if Vlayer > Hlayer
              [wrimage wm locimage Vlayer Hlayer hostsize payload] =
          extractVt2(wrimage,locimage,bv,i,block);
      if payload==0
                mesej = 'There do not have any watermarks inside this image';
          fprintf(ftxt,'\t\t%s\n',mesej);
          break;
      end
      masa=cputime-starttime;
      totaltime=totaltime+masa;
      psnr = rev_psnr(host,wrimage);
      [mssim, ssim_map] = ssim_index(host, wrimage);
      bitrate=payload/hostsize;
      tbitrate=tbitrate + bitrate;

          fprintf(ftxt,'\t%d\t\t%.2f\t%d\t\t%d\t\t\t%.4f\t%.4f\t%.4f\n',layer,masa,payload,host
          size,bitrate,psnr,mssim);
      if layer ==1
          fn3=char(wm);
          display 'Extracted text watermark :> '
          display '  '
```

```
          disp(fn3');
          fwrite(fout,fn3);
          fclose(fout);
       end
      else
       display 'start extractH'
       starttime=cputime;
              [wrimage wm locimage Vlayer Hlayer hostsize payload] =
          extractHt2(wrimage,locimage,bh,i,block);
       if payload==0
                 mesej = 'There do not have any watermarks inside this image';
          fprintf(ftxt,'\t\t%s\n',mesej);
          break;
       end
       masa=cputime-starttime;
       totaltime=totaltime+masa;
       psnr = rev_psnr(host,wrimage);
       [mssim, ssim_map] = ssim_index(host, wrimage);
       bitrate=payload/hostsize;
       tbitrate=tbitrate + bitrate;

fprintf(ftxt,'\t%d\t\t%.2f\t%d\t\t%d\t\t\t%.4f\t%.4f\t%.4f\n',layer,masa,payload,hostsize,bit
rate,psnr,mssim);
      end
      if layer > 1

imwrite(wm,strcat('c:\users\shahidan\documents\matlab\wm\','extwmLayer',num2str(layer
),'_',s1));
      end
      layer=layer-1;
   end
      fprintf(ftxt,'\t%s%.2f\t\t\t%s\t%.2f\n','Total time ',totaltime,'Total bitrate ',tbitrate);
      if s4==1
       imwrite(wrimage,strcat('c:\users\shahidan\documents\matlab\rev\','reversible_',s3));
       imwrite(locimage,strcat('c:\users\shahidan\documents\matlab\lokasi\','lokasi_',s3));
      else
       imwrite(wrimage,strcat('c:\users\shahidan\documents\matlab\rev\','reversible_',s1));
       imwrite(locimage,strcat('c:\users\shahidan\documents\matlab\lokasi\','lokasi_',s1));
      end
   totaltime=0;
   i=i+1;
end
fclose(ftxt);
```

Program 7 : Horizontal Extracting and Reconstructing

```
% extractV
function [wrimage w image3 nv nh hostSize wmSize] =
extractVt2(image2,image3,bv,cyc,block)

figure(1);
imshow(image2);       title('watermarked image');     % watermarking image
```

```
[row,col]=size(image2);
hostSize=row*col;
[ro,co]=size(image3);
wimage = image2;
wrimage = image2;
nv = double(image3(33,1));
nh = double(image3(34,1));
layer=nh+nv;
swm=layer*2;
if layer > 16
   wmSize=0;
   return;
end
wx = double(image3(swm-1,1));
wy = double(image3(swm,1));
if layer==1
   wx1 = double(image3(1,1));
   wx2 = double(image3(2,1));
    wx = 4*wx1 + wx2;
    wy = 1;
 end;
kes=0;

loc=dec2bin(image3);
round=1;
extract=0;
b1=bv;
b2=0;
i=1;
j=1;
neg=0;
w3=1;
rej=0;
boleh=0;
s=dec2bin(2);
ww=imread(strcat('c:\users\shahidan\documents\matlab\host\database\','lena.jpg'));
w=imresize(ww, [wx wy]);
wm=dec2bin(w);
[w1,w2] = size(wm)
wmSize=w1*w2
figure(2);
imshow(wimage);       title('before extract');

 i=1;
 j=1;
 b1=bv;
 b2=0;
while i < (row-block)
  while j<(col-block)
    r=0;
    c=0;
   while c <= block-1
    while r <= block-1-(block/2)
```

```
        b2=b2+1;
        if b2>8
            b1=b1+1;
            b2=1;
        end
        if boleh > wmSize
            round=3;
            i=row;
            j=col;
            r = block-1-(block/2);
            c = block-1;
            break;
        end
    kes=0;
    x1=double(image2(r+i,c+j));
    x2=double(image2(r+(block/2)+i,c+j));
    if(x1>x2)
        y=x1;
        x=x2;
        kes=1;
        else
         y=x2;
         x=x1;
         kes=2;
    end
     d=y-x;
     if loc(b1,b2)== s(2)
         extract=0;
         r=r+1;
       continue;
     end

if loc(b1,b2)== s(1)
  boleh=boleh+1;
  extract=1;

  msgbit = mod((y-x),2);
  f=floor((y-x)/2);
  yo=y-f-msgbit;
  if msgbit==0
      wm(w3)=s(2);
  else
      wm(w3)=s(1);
  end
  if kes==1
   wrimage(r+i,c+j)=yo;
  end
  if kes==2
   wrimage(r+(block/2)+i,c+j)=yo;
  end

end

r=r+1;
```

```
  if extract==1
      extract=0;
      w3=w3+1;        % goto next bit of wm
  end
  if(w3>wmSize)
       i=row;
       j=col;
       r=block-1-(block/2);
       c=block-1;
       break;
  end;

end;    % while r
   r=0;
   c=c+1;
end;   % while c
j=j+block;

 end;    % while j
   j=1;
   i=i+block;

end;    % while i

display 'extracted watermarked'
 if layer>1
   d4=bin2dec(wm);
   d5=reshape(d4,wx,wy);
   w(1:wx,1:wy) = d5(1:wx,1:wy);
   figure(layer*10+cyc);
   imshow(w);
 else
   w=bin2dec(wm);
 end
 nv=nv-1;
 image3(33,1)=nv;
end
```

Program 8 : Vertical Extracting and Reconstructing

```
% extractH
function [wrimage w image3 nv nh hostSize wmSize] =
extractHt2(image2,image3,bh,cyc,block)

figure(1);
imshow(image2);       title('watermarked image');     % watermarking image

[row,col]=size(image2);
hostSize=row*col;
```

```
[ro,co]=size(image3);
wimage = image2;
wrimage = image2;
nv = double(image3(33,1));
nh = double(image3(34,1));
layer=nh+nv;
if layer > 16
   wmSize=0;
   return;
end
swm=layer*2;
wx = double(image3(swm-1,1));
wy = double(image3(swm,1));

loc=dec2bin(image3);
[cw1,cw2]=size(loc);
round=1;
extract=0;
neg=0;
w3=1;
rej=0;
boleh=0;
s=dec2bin(2);
ww=imread(strcat('c:\users\shahidan\documents\matlab\host\database\','lena.jpg'));
w=imresize(ww,[wx wy]);
wm=dec2bin(w);
[w1,w2] = size(wm)
wmSize=w1*w2
figure(2);
imshow(wimage);       title('before extract');

 b1=bh;
 b2=0;
 i=row-block;
 j=col-block;
while i > block
  while j > block
     r=0;
     c=0;
    while r <= block-1
     while c <= block-1-(block/2)
        b2=b2+1;
        if b2>8
            b1=b1-1;
            b2=1;
        end
        if boleh > wmSize
            round=3;
            i=block;
            j=block;
            c = block-1-(block/2);
            r = block-1;
            break;
        end
```

```
 x1=double(image2(i-r,j-c));
 x2=double(image2(i-r,j-c-(block/2)));
 if(x1>x2)
     y=x1;
     x=x2;
     diff=y-x;
     kes=1;
 else
      y=x2;
      x=x1;
     diff=y-x;
     kes=2;
  end

  if loc(b1,b2)== s(2)
    c=c+1;
    continue;
  end

 if loc(b1,b2)== s(1)
  boleh=boleh+1;
  extract=1;
msgbit = mod((y-x),2);
 f=floor((y-x)/2);
yo=y-f-msgbit;
if msgbit==0
    wm(w3)=s(2);
else
    wm(w3)=s(1);
end
if kes==1
 wrimage(i-r,j-c)=yo;
else
 wrimage(i-r,j-c-(block/2))=yo;
end

end

 c=c+1;
if extract==1
    extract=0;
    w3=w3+1;   % goto next bit of wm
 end
 if(w3>wmSize)
     round=3;
     i=block;
     j=block;
     c=block-1-(block/2);
     r=block-1;
     break;
 end;

 end;    % while r
 c=0;
```

```
    r=r+1;
   end;   % while c
j=j-block;

 end;    % while j
    j=col-block;
    i=i-block;

end;    % while i

display 'extracted watermarked'
if layer>1
    d4=bin2dec(wm);
    d5=reshape(d4,wx,wy);
    w(1:wx,1:wy) = d5(1:wx,1:wy);
    figure(layer*10+cyc);
    imshow(w);
else
    w=bin2dec(wm);
end

watermarked image as a bmp file
image as a bmp file
nh=nh-1;
image3(34,1)=nh;
end
```

Program 9 : Check PSNR

```
function p = rev_psnar(image1,image2);


figure(1);
imshow(image1);       title('original image');
figure(2);
imshow(image2);       title('watermarked image');
[row,col] = size(image1)
size_host = row*col;
 o_double = double(image1);
 w_double = double(image2);
 s=0;
 for j = 1:size_host; % the size of the original image
 s = s+(w_double(j) - o_double(j))^2 ;
 end
 mes=s/size_host;
 psnr =10*log10((255)^2/mes);
 p=psnr;
```

Program 10 : Attack (Robustness)

```
clear all;close all;clc;
s3='lena.bmp';
wtr_image=imread(strcat('c:\users\shahidan\documents\matlab\hostWm\','wm_',s3));
(wtr_image,strcat('c:\users\shahidan\documents\matlab\hostWm\','wm_','lena.jpg'),'jpg','qu
ality',70);
%Attack_wtr_image = imread ('w_image.jpg');
%Attack_wtr_image=imnoise(wtr_image,'salt & pepper',0.02);
%Attack_wtr_image = imrotate(wtr_image,1);
%Attack_wtr_image = imrotate(Attack_wtr_image,-1);
%Attack_wtr_image = imcrop(Attack_wtr_image, [7 7 255 255]);
%Attack_wtr_image=medfilt2(wtr_image,[3 3]);
%Attack_wtr_image=wiener2(wtr_image,[3 3]);
Attack_wtr_image=imnoise(wtr_image,'gaussian',0.01);
imwrite(Attack_wtr_image,strcat('c:\users\shahidan\documents\matlab\hostWm\','wm_',s3
));
display 'tamat attack';
```

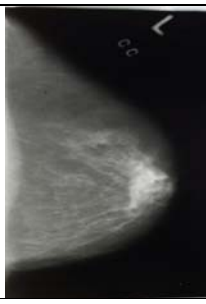
Program 11 : Attack (Determine NCC, BCR, BER and PSNR)

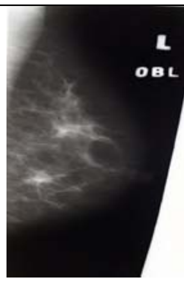
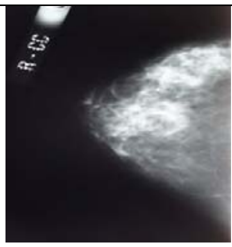
```
clear all;close all;clc;
s3='2_lena.jpg';
s4='_copyright.bmp';
wm=imread(strcat('c:\users\shahidan\documents\matlab\wm\extwmlayer',s3));
ori=imread(strcat('c:\users\shahidan\documents\matlab\wm\',s4));
ori16=uint16(ori);
wm16=uint16(wm);
atas = sum(sum(immultiply(ori16,wm16)));
bawah = sum(sum(immultiply(ori16,ori16)));
ncc = atas/bawah

%BCR

im1=imread(strcat('c:\users\shahidan\documents\matlab\wm\extwmlayer',s3));
im2=imread(strcat('c:\users\shahidan\documents\matlab\wm\',s4));

ixor = xor(im1,im2);
BER = sum(sum(ixor))/(50*300*8)*100
BCR=100-BER
```

APPENDIX B

Collection of Selected Additional 35 Host Images

1. Img1	2. Img2	3. Img3
		
4. Img4	5. Img5	6. Img6
		

7. Img7	8. Img8	9. Img9
		
10. Img10	11. Med1	12. Med2
		
13. Med3	14. Med4	15. Med5
		
16. Med6	17. Med7	18. Med8
		

19. Med9	20. Med10	21. Med11
		
22. Med12	23. Med13	24. Med14
		
25. Med15	26. Med16	27. Med17
		
28. Med18	29. Mil1	30. Mil2
		

31. Mil3	32. Mil4	33. Mil5
		
34. Mil6	35. Mil7	
		

APPENDIX C

Result of Embedding for Host Images

Multilayer Embedding the Watermark into 35 Host Image with Different block size: 2, 4, 6, 8, 10

1. im1		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.583	1	73.5598	1	73.6122	1	73.6122	1
2	0.0363	63.3388	0.9997	63.3361	0.9997	63.3339	0.9997	63.3388	0.9998	63.3388	0.9997
3	0.1521	58.4507	0.9991	55.5408	0.9983	54.8694	0.998	55.1715	0.9982	52.4068	0.9971
4	0.4573	49.5361	0.9948	46.0252	0.99	44.3607	0.9864	44.4377	0.9869	42.7046	0.9817
5	0.5862	47.7209	0.9923	44.2373	0.9847	42.9502	0.9813	42.9585	0.9819	41.1679	0.9741
6	0.7152	46.1641	0.9898	42.6708	0.979	41.711	0.9759	40.8287	0.9723	39.5717	0.9648
7	0.8441	43.2707	0.9808	40.4068	0.9649	39.6903	0.9623	38.5384	0.9535	37.1181	0.941
8	0.973	41.4803	0.9732	38.5171	0.9507	38.0485	0.9488	36.9064	0.9367	35.2757	0.9178
9	1.102	39.2906	0.9579	36.8547	0.9323	35.8482	0.9212	34.8948	0.9087	33.2743	0.8813
10	1.4653	35.2241	0.914	32.3898	0.8571	31.3277	0.8357	30.2861	0.8124	28.7267	0.768
11	1.8315	31.3824	0.8336	28.3225	0.741	26.6798	0.6915	25.3617	0.652		
12	2.1367	29.1194	0.7765								
2. im2		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.5888	1	73.6122	1	73.6122	1	73.6122	1	72.5988	1
2	0.0363	63.3366	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.233	0.9997
3	0.1521	58.6462	0.999	54.6489	0.9976	54.2032	0.9974	54.2498	0.9974	51.4454	0.9957
4	0.4573	47.979	0.9917	45.4306	0.9852	43.8568	0.9797	42.9131	0.977	41.3217	0.9691
5	0.5862	46.6165	0.989	43.7206	0.9792	42.4508	0.9741	41.6789	0.9714	40.0167	0.961

6	0.7152	45.3672	0.9857	42.1274	0.9714	40.3018	0.9598	39.847	0.9587	38.2773	0.9452
7	0.8441	43.9888	0.9817	40.898	0.9641	38.6159	0.9445	37.6926	0.937	36.6526	0.9252
8	0.973	41.7008	0.9702	38.7106	0.9439	36.8636	0.9213	35.5789	0.9055	34.393	0.8858
9	1.102	40.1802	0.9596	37.2435	0.9274	35.4632	0.9003	34.1299	0.8807	32.8983	0.8554
10	1.4653	35.2079	0.8933	32.3494	0.825	30.3558	0.7676	28.8423	0.728	27.4954	0.684
11	1.8315	32.368	0.8325	29.3194	0.7347	26.7884	0.641				
12	2.1367	29.3747	0.7358								
3. im3		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6122	1	73.6122	0.9999	73.5946	0.9999	73.5714	0.9999
2	0.0363	63.3388	0.9997	63.3388	0.9997	63.3388	0.9996	63.3372	0.9997	63.335	0.9996
3	0.1521	58.6484	0.9989	58.4933	0.9989	54.4346	0.9972	54.8402	0.9974	54.4806	0.9972
4	0.4573	48.3209	0.9911	46.8729	0.9879	44.6163	0.9808	43.953	0.9799	43.4276	0.9776
5	0.5862	46.8798	0.9879	45.707	0.9848	43.1144	0.975	42.5812	0.9739	42.1691	0.9723
6	0.7152	45.4957	0.9838	43.6035	0.9762	41.7056	0.9667	40.5099	0.9599	40.1962	0.9583
7	0.8441	44.219	0.9797	41.6433	0.9647	40.5701	0.9589	38.8679	0.9452	38.5886	0.9434
8	0.973	41.8902	0.9669	39.228	0.9422	38.5459	0.938	37.0913	0.9227	36.1341	0.9092
9	1.102	41.0265	0.9622	38.4402	0.9347	37.1282	0.9211	35.6313	0.9017	34.8963	0.8889
10	1.4653	35.6349	0.8904	32.824	0.8209	31.6783	0.7928	30.2411	0.7607	29.3347	0.7317
11	1.8315	33.0915	0.8373	29.9122	0.7278	28.4024	0.6813	27.0461	0.6418		
12	2.1367	29.7619	0.7293								

4. im4		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.5656	1	73.6122	1	73.6122	1	73.6122	1
2	0.0363	63.3388	0.9998	63.3344	0.9998	59.1951	0.9996	63.2918	0.9998	59.3456	0.9996
3	0.1521	50.4366	0.9978	47.4389	0.9958	45.8638	0.9942	46.0033	0.9944	44.6882	0.9927
4	0.4573	33.6483	0.9626	29.9867	0.9282	28.475	0.9113				
5	0.5862	33.0494	0.9576								
6	0.7152	31.9074	0.944								
7	0.8441	30.6762	0.9306								
8	0.973	28.7794	0.9001								
5. im5		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.5946	1	73.6122	1	73.6122	1	73.6122	1	73.6122	1
2	0.0363	63.3372	0.9998	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997
3	0.1521	55.1041	0.9981	54.1025	0.9977	51.2923	0.996	49.4886	0.9953	49.3141	0.9946
4	0.4573	41.7429	0.9821	39.2862	0.9711	38.3068	0.9633	37.1061	0.9578	36.4311	0.9508
5	0.5862	40.9243	0.9775	38.4392	0.9628	37.2154	0.9522	35.9721	0.9439	35.4029	0.9393
6	0.7152	39.4661	0.9678	37.279	0.9505	35.965	0.9376	34.9576	0.9304	34.2762	0.9227
7	0.8441	38.1304	0.9569	35.5301	0.9271	34.1747	0.9099	33.2188	0.9023	32.3728	0.8896
8	0.973	36.045	0.9328	33.8358	0.8995	32.4683	0.8784	31.5681	0.8701	30.7133	0.8541
9	1.102	34.916	0.9193	32.274	0.871	30.5904	0.839	29.7682	0.8291	28.9988	0.8139
10	1.4653	28.7202	0.7949	26.0632	0.7177	24.6504	0.6801				

6. im6		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	58.7194	0.9997	72.5988	1	72.4664	1	72.9611	1	68.0397	1
2	0.0363	48.9134	0.9974	58.6624	0.9997	56.1784	0.9995	56.2303	0.9994	55.9925	0.9994
3	0.1521	35.802	0.9692	45.9609	0.9949	44.6408	0.9934	44.6628	0.9934	43.6207	0.9919
4	0.4573	34.8125	0.9639	32.3142	0.9394	31.101	0.9261	30.5255	0.9199	30.1264	0.9142
5	0.5862	32.9242	0.9468	31.2286	0.9285	30.0901	0.9142	29.4965	0.9063	29.1691	0.901
6	0.7152	31.6263	0.934	29.6812	0.9058	28.512	0.8865				
7	0.8441	29.1966	0.8958								

7. im7		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6122	1	73.6122	1	73.6122	1	73.5598	1
2	0.0363	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.3339	0.9997
3	0.1521	58.5587	0.9991	55.876	0.9984	54.9994	0.9981	55.3494	0.9983	54.7083	0.998
4	0.4573	45.5844	0.9906	41.4745	0.9801	39.2538	0.9709	37.6546	0.9645	36.5993	0.9577
5	0.5862	44.6795	0.9884	40.7552	0.9757	38.7484	0.967	37.2805	0.9606	36.1428	0.9521
6	0.7152	43.0236	0.9818	39.3966	0.9643	37.6063	0.9534	36.2192	0.9452	35.229	0.9359
7	0.8441	41.3773	0.9747	38.213	0.9549	36.2922	0.9393	35.1996	0.9319	34.1349	0.9197
8	0.973	39.1935	0.9594	36.2031	0.929	33.9826	0.9016	33.1167	0.8923	32.3533	0.8811
9	1.102	37.7951	0.9495	34.7079	0.9111	32.784	0.8817	31.7665	0.8691	30.984	0.8546
10	1.4653	31.792	0.865	27.7472	0.7705	25.3321	0.7102				
11	1.8315	27.9865	0.772								

8 im8		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6122	1	73.6122	1	73.6122	1	73.5656	1
2	0.0363	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.3344	0.9997
3	0.1521	58.6484	0.999	58.6484	0.999	58.6484	0.9989	58.6484	0.9991	58.4854	0.999
4	0.4573	49.8127	0.9937	48.3774	0.9915	47.555	0.9901	46.3392	0.9888	45.9174	0.9877
5	0.5862	49.2682	0.9932	47.9411	0.9911	46.1979	0.9875	45.2193	0.9862	44.8668	0.9852
6	0.7152	47.2199	0.9894	46.3533	0.9875	43.823	0.9792	43.3407	0.9794	43.0293	0.9783
7	0.8441	46.4841	0.9887	44.756	0.9835	42.7777	0.9757	42.3981	0.9761	41.1968	0.9698
8	0.973	45.0995	0.9849	42.2517	0.9705	39.9004	0.9555	39.6918	0.9562	38.9776	0.9509
9	1.102	43.4785	0.98	41.2864	0.9661	38.773	0.9458	38.597	0.9484	37.517	0.9364
10	1.4653	38.5572	0.9402	35.6789	0.8927	33.1178	0.8377	32.6556	0.8377	31.8809	0.821
11	1.8315	35.7715	0.9036	32.811	0.8262	30.1336	0.7448	29.789	0.7506	28.2376	0.6976
12	2.1367	32.0872	0.81	29.0548	0.6976	26.3686	0.5973				
13	2.539	29.0306	0.7134								
9. im9		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	72.3293	1	73.6122	1	73.5656	1	73.5946	1	67.0761	1
2	0.0363	63.2009	0.9998	58.7294	0.9994	58.7243	0.9994	58.7376	0.9994	58.2793	0.9994
3	0.1521	51.519	0.9977	48.985	0.9957	47.4178	0.994	47.3524	0.9936	45.9934	0.9921
4	0.4573	40.9241	0.9819	36.5627	0.9546	34.7298	0.9369	33.728	0.9257	32.724	0.914
5	0.5862	39.607	0.9773	35.4577	0.9463	33.7366	0.9265	32.702	0.9134	31.7067	0.8996
6	0.7152	37.9686	0.9671	33.819	0.9243	32.1794	0.9007	31.0437	0.8821	30.1835	0.8684

7	0.8441	36.4167	0.9567	32.3357	0.9042	30.8698	0.8786	29.5771	0.8536	28.8024	0.8387
8	0.973	34.1948	0.9291	30.1222	0.8565	28.6952	0.8233				
9	1.102	33.0717	0.9158	28.9199	0.8305						
10	1.4653	27.6301	0.7967								
10. im10		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	72.7258	1	73.0682	1	68.0756	1	67.9833	1	68.0155	1
2	0.0363	58.7344	0.9999	56.1467	0.9997	56.0308	0.9998	54.1246	0.9996	54.0613	0.9996
3	0.1521	45.996	0.998	42.6864	0.9956	40.3339	0.9928	39.0565	0.9906	39.0271	0.9906
4	0.4573	33.6322	0.9773	29.4732	0.9463	27.3355	0.9222	25.5478	0.8977	24.9726	0.8893
5	0.5862	32.4207	0.9722								
6	0.7152	30.8429	0.9615								
7	0.8441	29.2731	0.9493								
11. med1		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6063	1	73.6122	1	73.6122	1	73.6122	1	73.6122	1
2	0.0363	63.3383	0.9997	63.3388	0.9996	63.3388	0.9996	63.3388	0.9997	63.3388	0.9996
3	0.1521	58.6482	0.9989	58.6484	0.9988	55.0682	0.9975	58.4578	0.9989	55.0313	0.9976
4	0.4573	49.6855	0.9932	49.4597	0.992	46.9445	0.9867	46.4019	0.9861	45.4015	0.9823
5	0.5862	47.9117	0.9895	47.6446	0.9884	45.8582	0.9835	45.3611	0.9827	43.794	0.9761
6	0.7152	46.3573	0.9855	46.058	0.9839	43.7168	0.9738	43.5042	0.9741	42.2571	0.967

7	0.8441	44.9772	0.9816	44.6383	0.9788	41.8075	0.9621	41.6905	0.9628	40.1014	0.9494
8	0.973	43.9019	0.9772	42.3631	0.9671	39.5313	0.9408	39.4128	0.94	38.3285	0.9273
9	1.102	42.7979	0.9716	40.7049	0.9536	38.0277	0.9219	37.9456	0.9219	36.6333	0.9014
10	1.4653	38.3934	0.9329	35.752	0.8822	32.9418	0.8092	32.916	0.8115	32.0026	0.7843
11	1.8315	35.3931	0.8834	32.3881	0.7897	29.7011	0.6901	29.4654	0.6897	28.25	0.643
12	2.1367	33.0042	0.8226	29.8046	0.6947						
13	2.539	29.314	0.6827								
12. med2		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6122	1	73.6122	1	73.6122	1	73.6122	1
2	0.0363	63.3388	0.9997	63.3388	0.9996	63.3388	0.9996	63.3388	0.9997	63.3388	0.9996
3	0.1521	58.6484	0.9989	58.6484	0.9988	54.8047	0.9974	58.5302	0.9989	54.8387	0.9975
4	0.4573	49.7512	0.9933	49.4518	0.9921	46.9178	0.9867	47.7402	0.9891	45.4158	0.9824
5	0.5862	47.9909	0.9898	47.6617	0.9886	45.8547	0.9836	46.3735	0.9857	43.7527	0.9762
6	0.7152	46.4726	0.9858	46.0914	0.9841	43.7213	0.9742	44.1225	0.9768	42.2286	0.9673
7	0.8441	45.0635	0.9819	44.6316	0.9789	41.8288	0.9625	42.0597	0.965	40.0596	0.9495
8	0.973	43.9469	0.9775	42.3677	0.9671	39.5299	0.9409	39.715	0.944	38.2911	0.9276
9	1.102	42.8289	0.9718	40.711	0.9536	38.032	0.9221	38.1498	0.9251	36.5544	0.9009
10	1.4653	38.3874	0.932	35.7762	0.8814	32.9469	0.8082	33.1408	0.8179	32.1716	0.7877
11	1.8315	35.4077	0.8823	32.6869	0.7977	29.6897	0.686	29.8297	0.6992	28.5704	0.652
12	2.1367	33.0403	0.8211	29.9525	0.6971						
13	2.539	29.3385	0.6783								
13. med3		Block 2		Block 4		Block 6		Block 8		Block 10	

Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.5946	1	73.6122	1	73.6122	1	73.6122	1
2	0.0363	63.3388	0.9997	63.3372	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997
3	0.1521	58.6484	0.9989	58.6479	0.9989	58.6484	0.9989	58.6484	0.9989	58.6028	0.9989
4	0.4573	50.0147	0.9937	49.7851	0.9926	48.0347	0.9896	48.2706	0.9903	46.6635	0.9869
5	0.5862	49.5016	0.993	47.9184	0.9892	46.6566	0.9864	46.7781	0.9869	45.5661	0.9838
6	0.7152	49.0428	0.9924	46.2724	0.9845	45.3811	0.9822	45.4655	0.9828	43.5083	0.975
7	0.8441	46.6318	0.9867	44.9527	0.9804	44.0657	0.9772	44.1954	0.9781	41.6307	0.9634
8	0.973	44.8247	0.98	42.5448	0.9679	41.8247	0.9634	41.9615	0.965	39.3003	0.9406
9	1.102	43.4552	0.9746	40.8285	0.9538	40.3129	0.9509	40.4048	0.9532	37.8742	0.9231
10	1.4653	38.9064	0.9365	36.017	0.8838	35.4751	0.8729	35.3669	0.8742	32.8175	0.8091
11	1.8315	35.773	0.8844	32.7775	0.7971	31.7483	0.7668	31.8038	0.777	28.8681	0.6687
12	2.1367	32.3127	0.7932	30.0194	0.6986	28.6591	0.6475	28.4849	0.6496		
13	2.539	29.094	0.6656	26.1934	0.5348						
14. med4		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.5656	1	73.6122	1	73.5888	1	73.4739	1	73.5946	1
2	0.0363	63.3344	0.9997	63.3388	0.9996	63.3322	0.9996	63.2875	0.9997	58.864	0.999
3	0.1521	54.3291	0.9978	53.9778	0.9975	51.4103	0.9958	51.3703	0.9956	50.8101	0.9949
4	0.4573	46.4805	0.9895	43.3576	0.9794	40.8159	0.966	39.1607	0.9543	38.7659	0.9501
5	0.5862	45.4231	0.9875	42.0018	0.9743	39.5276	0.9578	38.1934	0.9465	37.5376	0.9392
6	0.7152	43.2644	0.9798	39.9744	0.9602	37.9437	0.9423	36.5944	0.9273	35.6905	0.9129
7	0.8441	41.4443	0.9719	38.295	0.9462	36.3867	0.9233	35.0115	0.9045	34.392	0.8916

8	0.973	39.1383	0.954	35.9351	0.9143	34.2084	0.8815	32.9391	0.8577	32.021	0.8369
9	1.102	38.2766	0.9483	34.694	0.895	33.0899	0.8595	31.5073	0.8242	30.8134	0.8057
10	1.4653	34.0782	0.8841	29.3313	0.7482	27.7534	0.6893	26.1092	0.6356	25.2859	0.6072
11	1.8315	31.5525	0.8333								
12	2.1367	28.6357	0.7319								
15. med5		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.0682	1	73.6122	1	73.4739	1	72.5988	1	72.7258	1
2	0.0363	63.2848	0.9997	58.9124	0.9993	58.8991	0.9993	58.904	0.9992	58.8064	0.9992
3	0.1521	54.0095	0.998	50.9289	0.9959	50.8256	0.9958	48.9619	0.9936	48.9269	0.9936
4	0.4573	43.1049	0.9817	39.2516	0.9574	37.7596	0.9436	36.3342	0.9286	35.7505	0.9219
5	0.5862	41.8069	0.9774	38.2593	0.9506	36.7322	0.9343	35.272	0.916	34.7798	0.9093
6	0.7152	39.893	0.966	36.1669	0.925	34.719	0.903	33.4904	0.8835	33.2005	0.8795
7	0.8441	38.9998	0.9607	35.0146	0.9098	33.5909	0.8841	32.4061	0.8613	31.8463	0.8508
8	0.973	36.3825	0.9336	32.7262	0.863	31.2362	0.8259	30.2433	0.7973	29.5881	0.7806
9	1.102	35.3674	0.9222	31.8251	0.8446	30.1608	0.7986	29.2028	0.7682		
10	1.4653	30.1526	0.8057	26.2276	0.6575	24.5867	0.5944				
11	1.8315	28.312	0.7635								

16. med6		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.5888	1	73.6122	1	73.6122	1	73.5946	1
2	0.0363	63.3388	0.9997	63.3366	0.9997	63.3388	0.9996	63.3388	0.9997	63.3372	0.9997
3	0.1521	58.6484	0.9989	58.6477	0.9989	55.0887	0.9976	58.559	0.9989	54.9824	0.9977
4	0.4573	49.653	0.9933	48.0072	0.9904	45.789	0.9845	46.6032	0.9873	44.4771	0.9803
5	0.5862	47.8696	0.99	46.6348	0.9871	44.8993	0.9816	45.5095	0.9842	43.0901	0.9745
6	0.7152	46.2937	0.986	45.3914	0.9833	43.1087	0.9731	43.5785	0.9761	41.7529	0.9665
7	0.8441	44.9421	0.9824	44.1042	0.9787	41.4098	0.9625	41.697	0.9653	39.7763	0.9501
8	0.973	43.8487	0.9782	41.8713	0.9657	39.1775	0.9401	39.3859	0.9436	38.1076	0.9301
9	1.102	42.7021	0.9729	40.3529	0.954	37.7741	0.9231	37.9194	0.9268	36.4409	0.906
10	1.4653	38.3501	0.9358	35.36	0.8792	32.7035	0.8101	32.9149	0.8188	31.4249	0.7768
11	1.8315	35.3219	0.8881	32.1889	0.7957	29.5734	0.6997	29.4769	0.7012	27.7617	0.6368
12	2.1367	32.9453	0.8302	29.434	0.695						
13	2.539	29.2523	0.6954								
17. med7				Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate			PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6122	1	73.6122	1	73.6122	1	73.5714	1
2	0.0363	63.3388	0.9997	63.3388	0.9997	63.3388	0.9996	63.3388	0.9997	63.335	0.9996
3	0.1521	58.6484	0.9989	58.6484	0.9989	58.6406	0.9989	58.6484	0.999	58.478	0.9989
4	0.4573	49.7978	0.9934	49.4533	0.9924	47.5923	0.9891	47.6956	0.9897	47.404	0.989
5	0.5862	49.276	0.9928	47.6522	0.9893	46.3109	0.9861	46.3461	0.9866	46.1066	0.9861

6	0.7152	47.2653	0.9889	46.0545	0.985	44.0046	0.9773	45.1701	0.9833	43.7955	0.9774
7	0.8441	45.6012	0.9839	44.6848	0.9809	41.9985	0.9664	43.8773	0.9788	41.7938	0.9668
8	0.973	44.4415	0.9803	42.3368	0.9688	39.5473	0.9437	41.7206	0.9667	39.3958	0.946
9	1.102	43.146	0.9754	40.6762	0.9566	38.713	0.9369	40.1954	0.9565	37.8652	0.9282
10	1.4653	38.577	0.9355	35.6817	0.8822	33.2992	0.8265	35.221	0.8798	32.7349	0.8131
11	1.8315	35.6229	0.8892	32.7359	0.8049	30.3553	0.7223	31.4349	0.7717	29.0858	0.677
12	2.1367	32.0701	0.7911	30.0106	0.7062	27.2706	0.5863	28.3477	0.6448		
13	2.539	29.1162	0.6717	26.4612	0.5461						
18. med8		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6122	1	73.5888	1	73.5946	1	73.6122	1
2	0.0363	63.3388	0.9996	63.3388	0.9996	63.3366	0.9996	63.3372	0.9996	63.3388	0.9996
3	0.1521	58.6484	0.9989	58.6484	0.9988	58.4596	0.9988	58.6111	0.9989	54.7203	0.9973
4	0.4573	49.7851	0.9928	49.3826	0.9917	47.473	0.9879	47.5543	0.9883	46.5285	0.9856
5	0.5862	49.2348	0.9921	47.6505	0.9883	46.2272	0.9846	46.2191	0.9849	45.4705	0.9824
6	0.7152	48.8204	0.9915	46.0492	0.9837	43.9606	0.975	43.9853	0.9758	43.3953	0.9727
7	0.8441	46.5554	0.9865	44.5877	0.9786	41.9666	0.9632	41.9711	0.9644	41.5717	0.9621
8	0.973	44.7743	0.9796	42.2352	0.965	39.5398	0.9384	39.5839	0.9413	39.2747	0.9391
9	1.102	43.4097	0.9741	40.6267	0.952	38.6848	0.93	38.0487	0.9227	37.7974	0.9202
10	1.4653	38.7751	0.9317	35.6547	0.8718	34.3974	0.8424	33.0323	0.8069	32.665	0.7973
11	1.8315	35.9908	0.8862	32.6343	0.7873	31.11	0.7337	29.731	0.6831	29.0681	0.6596
12	2.1367	33.5579	0.824	29.9342	0.6814	28.0736	0.5974				
13	2.539	29.9819	0.6902								

19. med9		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6122	1	73.6063	1	73.6122	1	73.6063	1
2	0.0363	63.3388	0.9997	63.3388	0.9996	63.3383	0.9996	63.3388	0.9997	63.3383	0.9996
3	0.1521	58.6484	0.9989	58.6484	0.9988	58.6482	0.9988	58.6484	0.9989	58.6166	0.9988
4	0.4573	50.0169	0.9931	49.6851	0.992	47.9023	0.9886	48.0884	0.9893	47.6575	0.9882
5	0.5862	49.4778	0.9923	49.1476	0.9913	46.5612	0.9851	46.6574	0.9858	46.3141	0.9847
6	0.7152	49.0272	0.9916	47.0886	0.9862	45.3191	0.9807	45.3779	0.9814	43.9458	0.9747
7	0.8441	46.6175	0.986	45.3629	0.9806	44.0192	0.975	44.1478	0.9763	41.8993	0.9621
8	0.973	44.8207	0.9785	42.7106	0.9654	41.7819	0.9598	41.9044	0.9619	39.4547	0.9374
9	1.102	43.4399	0.9722	41.7006	0.9588	40.2557	0.9465	40.3574	0.9493	37.9464	0.9172
10	1.4653	38.8734	0.929	36.3112	0.8825	35.2527	0.858	35.1245	0.8591	32.8467	0.7932
11	1.8315	35.7611	0.8728	33.5021	0.8071	31.659	0.744	31.4683	0.7445	28.7939	0.6329
12	2.1367	32.196	0.7684	30.555	0.7023	28.4342	0.609	28.1768	0.607		
13	2.539	29.2273	0.6419	27.3113	0.5511						
20. med10		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6122	1	73.6122	1	73.6122	1	73.5888	1
2	0.0363	63.3388	0.9997	63.3388	0.9997	63.3388	0.9996	63.3388	0.9997	63.3366	0.9996
3	0.1521	58.6484	0.9989	58.6484	0.9989	58.6484	0.9988	58.6484	0.9989	58.4973	0.9988
4	0.4573	49.97	0.9933	49.7281	0.9923	47.9513	0.9891	47.1361	0.988	46.6751	0.9865
5	0.5862	49.4474	0.9926	47.8459	0.9888	46.5615	0.9856	45.9198	0.9845	45.5456	0.9831
6	0.7152	47.2934	0.9877	46.197	0.9838	45.333	0.9814	44.8405	0.9804	43.4494	0.9733

7	0.8441	45.5946	0.9822	44.8541	0.979	42.6934	0.9685	42.4594	0.9684	41.5439	0.961
8	0.973	44.412	0.9776	42.451	0.9654	40.9298	0.9537	40.8205	0.9548	39.1818	0.9363
9	1.102	43.1259	0.9715	40.7358	0.9513	39.8865	0.9437	39.8057	0.9453	37.7138	0.9165
10	1.4653	38.6328	0.9294	35.8197	0.8741	34.8429	0.8532	34.3884	0.8471	32.4121	0.7884
11	1.8315	35.6642	0.8787	32.537	0.7816	31.0582	0.7326	30.6136	0.7235	28.3565	0.6337
12	2.1367	32.1432	0.7826	29.7501	0.6778	27.9194	0.6042	27.4701	0.5923		
13	2.539	29.1261	0.6593								
21. med11		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.5946	1	73.583	1	73.6122	1	73.5656	1	73.5946	1
2	0.0363	63.3372	0.9997	63.3361	0.9997	63.3388	0.9997	63.3344	0.9997	63.3372	0.9997
3	0.1521	58.6479	0.9989	58.6475	0.9989	58.3972	0.9989	58.5869	0.999	54.9487	0.9976
4	0.4573	49.9193	0.9936	49.518	0.9927	47.6293	0.9895	47.7629	0.9901	45.5148	0.9842
5	0.5862	48.0844	0.9905	47.7421	0.9897	46.3412	0.9866	46.3894	0.9871	44.6061	0.9815
6	0.7152	47.6839	0.9901	46.1434	0.9857	45.2157	0.9833	45.2499	0.9841	42.8875	0.9744
7	0.8441	45.9106	0.9863	44.6988	0.9815	42.6663	0.9723	42.7178	0.9738	41.2124	0.9649
8	0.973	44.2621	0.9805	42.3521	0.9704	40.9413	0.9602	40.9701	0.9623	39.0332	0.9441
9	1.102	43.0872	0.9761	40.691	0.9586	38.8929	0.9409	38.9013	0.9439	37.6246	0.9286
10	1.4653	38.6938	0.9398	35.7336	0.888	34.4973	0.8625	34.5373	0.8702	32.5836	0.8187
11	1.8315	35.5605	0.8932	32.3342	0.8006	30.1889	0.7262	30.2983	0.7418	28.6767	0.6778
12	2.1367	32.027	0.7979	29.6794	0.7041	27.186	0.5977	27.2496	0.6147		
13	2.539	28.5169	0.659								

22. med12		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	72.9611	1	73.6122	1	73.6122	1	73.6122	1
2	0.0363	63.3388	0.9997	63.2734	0.9996	63.3388	0.9996	63.3388	0.9997	63.3388	0.9997
3	0.1521	58.6484	0.9989	58.6261	0.9988	58.3993	0.9988	58.5492	0.9989	54.7524	0.9974
4	0.4573	50.0036	0.9936	49.7016	0.9925	47.7901	0.989	46.5372	0.9867	45.4316	0.9828
5	0.5862	48.1153	0.9902	47.8104	0.9891	46.4408	0.9859	45.4488	0.9836	43.7663	0.9768
6	0.7152	46.5804	0.9863	46.2312	0.9848	44.2329	0.9773	43.6286	0.9756	42.3162	0.9687
7	0.8441	45.1321	0.9824	44.8324	0.9799	42.1016	0.9654	41.7434	0.9647	40.1128	0.9514
8	0.973	44.017	0.978	42.4883	0.9678	39.7575	0.9445	39.5087	0.9432	38.3778	0.9306
9	1.102	42.9512	0.9727	40.7969	0.9553	38.1677	0.9257	38.0039	0.9258	36.6494	0.9051
10	1.4653	38.5399	0.934	35.8496	0.8837	33.1212	0.8167	32.7532	0.8118	31.7083	0.7796
11	1.8315	35.158	0.8786	32.4309	0.795	29.4764	0.6876	29.1085	0.6849	27.8318	0.6345
12	2.1367	31.8224	0.7868	29.6549	0.6942						
13	2.539	28.4244	0.6508								
23. med13		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.5946	1	73.6122	1	73.6122	1	73.6122	1	73.6005	1
2	0.0363	63.3372	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.3377	0.9997
3	0.1521	58.6479	0.999	58.5517	0.9989	54.7989	0.9975	55.3278	0.9978	54.7575	0.9975
4	0.4573	49.7413	0.9938	49.3735	0.9925	46.7802	0.9873	45.7785	0.9853	45.2362	0.983
5	0.5862	47.9617	0.9904	47.6111	0.9893	45.6607	0.9843	44.8538	0.9823	43.5903	0.977

6	0.7152	46.4096	0.9866	46.0498	0.9852	43.545	0.9756	43.1121	0.9747	42.073	0.9687
7	0.8441	44.9969	0.9829	43.1669	0.9733	41.6733	0.9648	41.4192	0.9647	39.9307	0.9518
8	0.973	43.898	0.9788	41.4547	0.962	39.3871	0.9449	39.204	0.9437	38.1566	0.9308
9	1.102	41.7728	0.9669	40.3843	0.9546	37.8996	0.9268	37.7834	0.9275	36.4327	0.9058
10	1.4653	38.0024	0.9306	35.7577	0.8914	32.8245	0.8195	32.7211	0.819	31.8582	0.7913
11	1.8315	34.1775	0.8571	32.3094	0.8001	29.3313	0.6902	29.323	0.699	27.8349	0.6385
12	2.1367	31.2939	0.77	29.7396	0.7085						
13	2.539	27.3839	0.6023								
24. med14		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.5946	1	73.6122	1	73.6122	1	73.6122	1	73.6122	1
2	0.0363	63.3372	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997
3	0.1521	58.6479	0.9989	54.865	0.9976	54.5851	0.9974	54.918	0.9976	54.5747	0.9974
4	0.4573	49.5296	0.9934	47.054	0.9886	45.2928	0.9836	45.313	0.9839	44.8422	0.9817
5	0.5862	47.8143	0.9903	45.9355	0.9858	43.6319	0.9776	43.6227	0.9781	43.275	0.9761
6	0.7152	46.2551	0.9868	44.9216	0.9828	42.129	0.9697	42.166	0.9707	40.8618	0.96
7	0.8441	44.7598	0.9827	42.5716	0.9722	39.9536	0.9528	40.0075	0.9549	39.0757	0.9437
8	0.973	43.7245	0.979	40.8443	0.9601	38.1745	0.932	38.2418	0.9352	37.1807	0.9203
9	1.102	41.6549	0.9681	38.8912	0.9419	36.4792	0.9078	36.5363	0.912	35.7406	0.8975
10	1.4653	37.8792	0.9326	34.7219	0.872	32.1219	0.8015	31.9767	0.8039	30.6714	0.76
11	1.8315	34.0012	0.8613	31.0561	0.7683	28.5328	0.6689	28.2873	0.671	27.0958	0.6195
12	2.1367	31.1364	0.776	28.6736	0.677						
13	2.539	27.2252	0.6117								

25. med15		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6122	1	73.6122	1	73.6122	1	73.6122	1
2	0.0363	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997
3	0.1521	58.6484	0.9989	58.6484	0.9989	58.6068	0.9989	58.6484	0.999	58.4585	0.9989
4	0.4573	49.7795	0.9935	48.3159	0.991	47.692	0.9896	46.7415	0.9882	46.4427	0.9874
5	0.5862	49.2605	0.9929	46.8688	0.988	46.3472	0.9867	45.6176	0.9853	45.3545	0.9846
6	0.7152	47.2012	0.9889	45.5242	0.9841	43.9984	0.9779	43.6513	0.9778	43.3342	0.9765
7	0.8441	45.5054	0.9841	44.281	0.9802	41.9335	0.967	41.763	0.9681	41.4733	0.9663
8	0.973	44.3331	0.9805	41.9496	0.9671	39.5102	0.9452	39.4323	0.9468	39.1541	0.945
9	1.102	43.0365	0.9756	40.3899	0.9557	37.9951	0.9276	38.6387	0.9405	37.6852	0.9281
10	1.4653	38.5533	0.9375	35.4037	0.88	32.9763	0.8187	33.2679	0.8346	32.3795	0.8091
11	1.8315	35.5939	0.8926	32.2887	0.7974	29.422	0.6904	29.9117	0.7215	28.3322	0.66
12	2.1367	32.0863	0.8013	29.5677	0.6995						
13	2.539	28.8327	0.6753								
26. med16		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6063	1	73.6122	1	73.6122	1	73.6122	1
2	0.0363	63.3388	0.9997	63.3383	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997
3	0.1521	58.6484	0.9989	58.6482	0.9989	58.6484	0.9989	58.6484	0.999	58.5488	0.999
4	0.4573	49.8749	0.9935	48.3617	0.991	46.7185	0.9875	45.6566	0.9854	45.5276	0.9847
5	0.5862	49.3592	0.9929	47.0361	0.9881	45.6981	0.9848	44.8313	0.9827	44.6772	0.982
6	0.7152	47.4064	0.9891	45.8062	0.9845	43.7011	0.9762	43.2156	0.975	43.0865	0.9744

7	0.8441	45.6864	0.9843	44.6402	0.9807	41.9334	0.9662	41.6257	0.966	41.4215	0.9646
8	0.973	44.5674	0.9806	42.2661	0.9676	39.552	0.9441	39.4745	0.946	38.6006	0.9362
9	1.102	43.2858	0.9755	40.6749	0.9564	38.1258	0.9278	38.0919	0.9311	36.891	0.9129
10	1.4653	38.7678	0.939	35.3813	0.88	32.783	0.816	32.2763	0.81	30.8641	0.7659
11	1.8315	34.8913	0.8771	31.1457	0.7678	28.3281	0.6683	27.9259	0.6653	26.1724	0.5999
12	2.1367	31.8067	0.7982	28.1489	0.6591						
13	2.539	27.6364	0.6458								
27. med17		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.5024	1	73.5946	1	73.6122	1	73.6122	1	73.5888	1
2	0.0363	63.3284	0.9997	63.3372	0.9996	63.3388	0.9996	63.3388	0.9996	63.3366	0.9996
3	0.1521	58.6449	0.9989	58.6479	0.9988	58.6484	0.9988	58.6484	0.9989	58.632	0.9988
4	0.4573	50.0475	0.9931	49.7307	0.992	48.0059	0.9888	48.1418	0.9894	47.7133	0.9884
5	0.5862	49.5212	0.9923	49.2033	0.9912	46.6551	0.9854	46.7242	0.986	46.3748	0.985
6	0.7152	49.064	0.9917	47.1618	0.9863	45.3959	0.981	45.442	0.9816	43.9985	0.975
7	0.8441	48.126	0.9902	45.4155	0.9808	44.0991	0.9755	44.1786	0.9765	41.9467	0.963
8	0.973	45.6443	0.9827	42.7932	0.9658	41.8495	0.9601	41.9435	0.9622	39.5176	0.9388
9	1.102	43.9803	0.9752	41.778	0.9594	40.3173	0.9475	40.3926	0.9502	38.0032	0.9196
10	1.4653	38.9796	0.9295	36.4183	0.8839	35.3354	0.8595	35.2013	0.8619	32.6923	0.7922
11	1.8315	36.045	0.879	33.5831	0.8098	31.7814	0.7502	31.5596	0.7512	28.9137	0.6465
12	2.1367	32.392	0.7755	30.6469	0.7055	28.5444	0.6144	28.2687	0.6149		
13	2.539	29.3428	0.6519	27.1388	0.547						

28. med18		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6122	1	73.5946	1	73.6122	1	73.6122	1
2	0.0363	63.3388	0.9997	63.3388	0.9997	63.3372	0.9996	63.3388	0.9997	63.3388	0.9996
3	0.1521	58.6484	0.9988	58.6484	0.9988	58.6479	0.9988	58.6484	0.9988	58.6331	0.9988
4	0.4573	50.004	0.9928	49.746	0.9919	48.1001	0.9888	48.2632	0.9894	47.8014	0.9883
5	0.5862	49.4716	0.992	49.2066	0.9911	46.681	0.9848	46.7774	0.9854	46.4246	0.9844
6	0.7152	47.3548	0.987	47.1118	0.9856	45.4014	0.9799	45.4569	0.9805	44.0009	0.9737
7	0.8441	45.6449	0.9808	45.3274	0.9791	44.0749	0.9738	44.1829	0.9749	41.934	0.9601
8	0.973	44.4254	0.9754	42.6975	0.9642	41.8077	0.9577	41.9127	0.9597	39.4673	0.9353
9	1.102	43.1107	0.9685	41.6768	0.9571	40.2539	0.9431	40.3508	0.9458	37.9456	0.9143
10	1.4653	38.6284	0.9223	36.3714	0.8813	35.0753	0.8485	35.0164	0.8522	32.6816	0.7868
11	1.8315	35.4429	0.8626	33.2326	0.7944	31.0114	0.7163	30.8024	0.7174	28.5787	0.6296
12	2.1367	32.0905	0.7631	30.2329	0.6861	27.6971	0.5787	27.4709	0.5793		
13	2.539	28.5817	0.6203	26.2988	0.512						
29. mil1		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6122	1	73.6122	1	73.6122	1	73.6122	1
2	0.0363	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997
3	0.1521	58.6484	0.999	58.6484	0.999	58.6484	0.999	58.6484	0.9991	58.6484	0.9991
4	0.4573	50.6915	0.9946	50.5335	0.9943	50.2114	0.9939	49.6424	0.9937	49.1231	0.9933
5	0.5862	50.1234	0.9942	49.9645	0.9938	49.688	0.9934	49.1665	0.9933	48.6993	0.993

6	0.7152	49.5799	0.9938	47.6495	0.9896	47.4515	0.9895	47.1507	0.9897	46.828	0.9897
7	0.8441	48.6321	0.9931	46.9938	0.9893	45.5931	0.9851	46.5612	0.9896	45.1358	0.986
8	0.973	45.9666	0.9871	45.255	0.9847	44.2396	0.981	44.941	0.9856	42.4998	0.9751
9	1.102	44.0257	0.9805	43.551	0.9786	42.9379	0.9764	43.3024	0.9804	41.5519	0.9717
10	1.4653	39.2445	0.9434	38.7448	0.9379	37.9943	0.9302	37.8456	0.9356	35.8783	0.9059
11	1.8315	36.2808	0.9019	35.0236	0.873	33.7936	0.8456	34.1761	0.8662	32.0215	0.8091
12	2.1367	32.7155	0.8131	31.8449	0.7823	30.5116	0.7381	30.3773	0.745	28.091	0.6655
13	2.539	29.5884	0.7083	28.035	0.6382	26.3107	0.565	26.1991	0.5722		
30. mil2		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.5888	1	73.6122	1	73.5252	1	73.6122	1	73.6122	1
2	0.0363	63.3366	0.9997	63.3388	0.9997	63.2956	0.9997	63.2918	0.9997	59.0127	0.9994
3	0.1521	54.2024	0.9978	51.6583	0.9962	51.4257	0.9961	49.4611	0.9945	49.0124	0.994
4	0.4573	39.1893	0.9784	36.6277	0.9626	36.0724	0.9579	34.7869	0.9507	34.231	0.945
5	0.5862	38.6359	0.9745	35.8798	0.9534	35.3659	0.9493	34.0604	0.9394	33.4078	0.9309
6	0.7152	37.6634	0.9641	34.8865	0.9377	34.2621	0.9298	32.9928	0.9177	32.3596	0.9081
7	0.8441	36.1438	0.946	33.4456	0.9132	32.6988	0.9021	31.493	0.8874	30.8215	0.8754
8	0.973	34.2135	0.9169	31.695	0.8772	30.9293	0.8632	29.9061	0.8488	29.2078	0.8342
9	1.102	33.1325	0.9016	30.067	0.8434	29.3764	0.8295				
10	1.4653	26.3848	0.7651	23.5501	0.6829						

31. mil3		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6122	1	73.6122	1	73.6122	1	73.6122	1
2	0.0363	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997
3	0.1521	55.9139	0.9988	55.5754	0.9985	53.2852	0.9979	53.3185	0.9978	53.0345	0.9976
4	0.4573	41.8931	0.9853	38.1273	0.9691	35.5593	0.9534	33.7732	0.9421	32.4094	0.9316
5	0.5862	41.0821	0.9828	37.4895	0.9652	34.8333	0.9463	33.2661	0.935	31.9273	0.923
6	0.7152	39.5761	0.9737	36.0014	0.9492	33.794	0.9297	32.1232	0.91	30.9476	0.8973
7	0.8441	38.2599	0.9666	34.7183	0.9364	32.5523	0.9116	30.9871	0.8909	29.8757	0.8752
8	0.973	35.3747	0.935	32.295	0.896	30.4738	0.868	28.8226	0.8382		
9	1.102	33.9756	0.9219	30.5907	0.871	28.8444	0.8372				
10	1.4653	27.6212	0.8001	23.6915	0.7078						
32. mil4		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6122	1	73.6122	1	73.6122	1	73.6063	1
2	0.0363	63.3388	0.9997	63.3388	0.9996	63.3388	0.9996	63.3388	0.9997	63.3383	0.9996
3	0.1521	58.6484	0.9989	54.7008	0.9975	54.3562	0.9973	54.5297	0.9974	54.264	0.9972
4	0.4573	48.2517	0.9915	45.7246	0.9855	44.183	0.9803	42.822	0.9768	42.4967	0.975
5	0.5862	46.7404	0.9888	44.7857	0.983	42.7259	0.9753	41.6616	0.9719	41.4043	0.9705
6	0.7152	45.3684	0.9852	42.9135	0.9748	40.4806	0.9605	39.8756	0.9591	39.6673	0.9578
7	0.8441	43.9887	0.9814	41.1394	0.9658	38.7285	0.9453	38.3036	0.9455	37.5227	0.9365
8	0.973	41.6426	0.9691	38.8557	0.9446	36.9181	0.9221	36.0022	0.9146	35.4631	0.9054
9	1.102	40.063	0.9586	37.3797	0.9295	35.4775	0.9017	34.7683	0.8955	34.0204	0.8803

10	1.4653	34.9556	0.8857	31.9508	0.8132	30.0608	0.7595	28.8747	0.7355	28.2903	0.716
11	1.8315	31.4257	0.7985	28.3513	0.6979	26.0537	0.6184				
12	2.1367	28.0649	0.6833								
33. mil5		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6122	1	73.5888	1	73.6122	1	73.5946	1
2	0.0363	63.3388	0.9998	63.3388	0.9997	63.3366	0.9997	63.3388	0.9998	63.3372	0.9997
3	0.1521	58.6484	0.999	58.6484	0.999	55.3472	0.9981	55.982	0.9982	54.9925	0.998
4	0.4573	48.3788	0.9932	45.8826	0.9894	44.1557	0.9842	42.5718	0.9812	42.3782	0.9797
5	0.5862	46.9456	0.9904	45.0274	0.9867	42.9847	0.979	41.7221	0.9761	41.4759	0.9753
6	0.7152	45.5157	0.9868	43.2625	0.9794	41.7627	0.9728	40.8225	0.9704	39.8516	0.9645
7	0.8441	44.2496	0.9836	41.6103	0.9705	39.8584	0.9587	39.3215	0.9578	38.4795	0.953
8	0.973	41.9294	0.9732	39.26	0.9518	38.156	0.9428	37.8238	0.943	36.9509	0.9349
9	1.102	40.303	0.9633	37.792	0.937	36.4976	0.9224	36.2938	0.9246	35.0502	0.9082
10	1.4653	35.4096	0.9058	32.2369	0.8359	30.9869	0.8026	30.394	0.7975	29.5392	0.7731
11	1.8315	31.7227	0.8252	28.1939	0.7061	26.6147	0.6555	26.1653	0.6495		
12	2.1367	28.505	0.7333								
34. mil6		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6122	1	73.6122	1	73.6122	1	73.6122	1	73.6122	1
2	0.0363	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997	63.3388	0.9997
3	0.1521	58.6484	0.999	58.6484	0.999	58.6484	0.999	58.6484	0.9991	58.6484	0.9991
4	0.4573	48.9518	0.9935	47.6272	0.9916	47.3948	0.991	46.576	0.9903	46.2795	0.9896

5	0.5862	48.5629	0.9931	47.3261	0.9912	46.3017	0.9885	46.3296	0.9899	45.385	0.9873
6	0.7152	46.7476	0.9893	45.9142	0.9875	45.0843	0.9849	45.1758	0.9866	43.5212	0.9806
7	0.8441	45.0688	0.9848	44.4384	0.983	43.9646	0.9817	43.9055	0.9828	42.7311	0.9787
8	0.973	42.4258	0.9727	42.1131	0.9711	41.7617	0.9703	41.7597	0.9719	39.9324	0.9606
9	1.102	41.5605	0.9692	41.2474	0.9675	40.2837	0.9613	40.9122	0.9685	38.8767	0.9528
10	1.4653	36.0666	0.9087	35.1924	0.8936	34.6403	0.881	34.4834	0.8888	32.8218	0.8511
11	1.8315	33.5041	0.8557	32.1905	0.8195	30.9069	0.7779	31.1069	0.7963	29.4083	0.7431
12	2.1367	30.309	0.7653	28.7055	0.7065	27.333	0.6496	27.3168	0.6618		
13	2.539	27.3793	0.6587								
35. mil7		Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	total bitrate	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
1	0.0058	73.6063	1	73.5946	1	73.5946	1	73.6122	1	73.6122	1
2	0.0363	63.3383	0.9997	63.3372	0.9997	63.3372	0.9997	63.3388	0.9997	63.3388	0.9997
3	0.1521	58.6482	0.999	58.6479	0.999	58.6479	0.9989	58.6484	0.9991	58.6484	0.999
4	0.4573	50.7934	0.9947	50.3291	0.9939	49.5243	0.9929	49.8804	0.9935	47.7141	0.991
5	0.5862	50.2499	0.9942	49.7945	0.9934	47.8147	0.9901	47.9906	0.9907	46.4167	0.9882
6	0.7152	49.7345	0.9938	49.3019	0.9929	46.2392	0.9862	46.3215	0.987	45.2386	0.9853
7	0.8441	48.7855	0.9929	48.3954	0.992	44.7642	0.9824	44.9145	0.9837	43.9056	0.9813
8	0.973	46.1047	0.9868	45.8235	0.986	43.6356	0.9784	42.5873	0.9746	41.7375	0.9707
9	1.102	44.3586	0.9805	44.1551	0.9798	42.5307	0.9735	40.8768	0.9643	40.2908	0.9624
10	1.4653	39.4248	0.9427	39.1607	0.9396	38.2604	0.9352	35.8742	0.8996	35.2309	0.8925
11	1.8315	36.7034	0.904	36.4942	0.9015	34.7871	0.876	32.87	0.8287	31.9992	0.811
12	2.1367	34.26	0.8484	32.7167	0.8007	31.6806	0.7843	30.2838	0.7423	28.529	0.6772
13	2.539	30.8507	0.7382	29.8253	0.6888	28.0289	0.6287	26.7796	0.5877		

APPENDIX D

Results of Extracting and Reconstructing

Multilayer Extracting and Reconstructing the Watermark with Different block size: 2, 4, 6, 8, 10

1. im1	Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
12	31.3824	0.8336	32.3898	0.8571	31.3277	0.8357	30.2861	0.8124	33.2743	0.8813
11	35.2241	0.914	36.8547	0.9323	35.8482	0.9212	34.8948	0.9087	35.2757	0.9178
10	39.2906	0.9579	38.5171	0.9507	38.0485	0.9488	36.9064	0.9367	37.1181	0.941
9	41.4803	0.9732	40.4068	0.9649	39.6903	0.9623	38.5384	0.9535	39.5717	0.9648
8	43.2707	0.9808	42.6708	0.979	41.711	0.9759	40.8287	0.9723	41.1679	0.9741
7	46.1641	0.9898	44.2373	0.9847	42.9502	0.9813	42.9585	0.9819	42.7046	0.9817
6	47.7209	0.9923	46.0252	0.99	44.3607	0.9864	44.4377	0.9869	52.4068	0.9971
5	49.5361	0.9948	55.5408	0.9983	54.8694	0.998	55.1715	0.9982	63.3388	0.9997
4	58.4507	0.9991	63.3361	0.9997	63.3339	0.9997	63.3388	0.9998	73.6122	1
3	63.3388	0.9997	73.583	1	73.5598	1	73.6122	1	Inf	1
2	73.6122	1	Inf	1	Inf	1	Inf	1		
1	Inf	1								
2. im2	Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
12	32.368	0.8325	32.3494	0.825	30.3558	0.7676	34.1299	0.8807	32.8983	0.8554
11	35.2079	0.8933	37.2435	0.9274	35.4632	0.9003	35.5789	0.9055	34.393	0.8858
10	40.1802	0.9596	38.7106	0.9439	36.8636	0.9213	37.6926	0.937	36.6526	0.9252
9	41.7008	0.9702	40.898	0.9641	38.6159	0.9445	39.847	0.9587	38.2773	0.9452
8	43.9888	0.9817	42.1274	0.9714	40.3018	0.9598	41.6789	0.9714	40.0167	0.961
7	45.3672	0.9857	43.7206	0.9792	42.4508	0.9741	42.9131	0.977	41.3217	0.9691
6	46.6165	0.989	45.4306	0.9852	43.8568	0.9797	54.2498	0.9974	51.4454	0.9957

5	47.979	0.9917	54.6489	0.9976	54.2032	0.9974	63.3388	0.9997	63.233	0.9997
4	58.6462	0.999	63.3388	0.9997	63.3388	0.9997	73.6122	1	72.5988	1
3	63.3366	0.9997	73.6122	1	73.6122	1	Inf	1	Inf	1
2	73.5888	1	Inf	1	Inf	1				
1	Inf	1								
3. im3	Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
12	33.0915	0.8373	32.824	0.8209	31.6783	0.7928	30.2411	0.7607	34.8963	0.8889
11	35.6349	0.8904	38.4402	0.9347	37.1282	0.9211	35.6313	0.9017	36.1341	0.9092
10	41.0265	0.9622	39.228	0.9422	38.5459	0.938	37.0913	0.9227	38.5886	0.9434
9	41.8902	0.9669	41.6433	0.9647	40.5701	0.9589	38.8679	0.9452	40.1962	0.9583
8	44.219	0.9797	43.6035	0.9762	41.7056	0.9667	40.5099	0.9599	42.1691	0.9723
7	45.4957	0.9838	45.707	0.9848	43.1144	0.975	42.5812	0.9739	43.4276	0.9776
6	46.8798	0.9879	46.8729	0.9879	44.6163	0.9808	43.953	0.9799	54.4806	0.9972
5	48.3209	0.9911	58.4933	0.9989	54.4346	0.9972	54.8402	0.9974	63.335	0.9996
4	58.6484	0.9989	63.3388	0.9997	63.3388	0.9996	63.3372	0.9997	73.5714	0.9999
3	63.3388	0.9997	73.6122	1	73.6122	0.9999	73.5946	0.9999	Inf	1
2	73.6122	1	Inf	1	Inf	1	Inf	1		
1	Inf	1								
4. im4	Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
8	30.6762	0.9306	47.4389	0.9958	45.8638	0.9942	63.2918	0.9998	59.3456	0.9996
7	31.9074	0.944	63.3344	0.9998	59.1951	0.9996	73.6122	1	73.6122	1
6	33.0494	0.9576	73.5656	1	73.6122	1	Inf	1	Inf	1

5	33.6483	0.9626	Inf	1	Inf	1				
4	50.4366	0.9978								
3	63.3388	0.9998								
2	73.6122	1								
1	Inf	1								
5. im5	Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
10	34.916	0.9193	32.274	0.871	30.5904	0.839	31.5681	0.8701	30.7133	0.8541
9	36.045	0.9328	33.8358	0.8995	32.4683	0.8784	33.2188	0.9023	32.3728	0.8896
8	38.1304	0.9569	35.5301	0.9271	34.1747	0.9099	34.9576	0.9304	34.2762	0.9227
7	39.4661	0.9678	37.279	0.9505	35.965	0.9376	35.9721	0.9439	35.4029	0.9393
6	40.9243	0.9775	38.4392	0.9628	37.2154	0.9522	37.1061	0.9578	36.4311	0.9508
5	41.7429	0.9821	39.2862	0.9711	38.3068	0.9633	49.4886	0.9953	49.3141	0.9946
4	55.1041	0.9981	54.1025	0.9977	51.2923	0.996	63.3388	0.9997	63.3388	0.9997
3	63.3372	0.9998	63.3388	0.9997	63.3388	0.9997	73.6122	1	73.6122	1
2	73.5946	1	73.6122	1	73.6122	1	Inf	1	Inf	1
1	Inf	1	Inf	1	Inf	1				
6. im6	Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
8	31.6263	0.934	31.2286	0.9285	30.0901	0.9142	30.5255	0.9199	30.1264	0.9142
7	32.9242	0.9468	32.3142	0.9394	31.101	0.9261	44.6628	0.9934	43.6207	0.9919
6	34.8125	0.9639	45.9609	0.9949	44.6408	0.9934	56.2303	0.9994	55.9925	0.9994
5	35.802	0.9692	58.6624	0.9997	56.1784	0.9995	72.9611	1	68.0397	1
4	48.9134	0.9974	72.5988	1	72.4664	1	Inf	1	88.3368	1

3	58.7194	0.9997	Inf	1	Inf	1				
2	72.7258	1								
1	Inf	1								
7. im7	Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
11	31.792	0.865	34.7079	0.9111	32.784	0.8817	33.1167	0.8923	32.3533	0.8811
10	37.7951	0.9495	36.2031	0.929	33.9826	0.9016	35.1996	0.9319	34.1349	0.9197
9	39.1935	0.9594	38.213	0.9549	36.2922	0.9393	36.2192	0.9452	35.229	0.9359
8	41.3773	0.9747	39.3966	0.9643	37.6063	0.9534	37.2805	0.9606	36.1428	0.9521
7	43.0236	0.9818	40.7552	0.9757	38.7484	0.967	37.6546	0.9645	36.5993	0.9577
6	44.6795	0.9884	41.4745	0.9801	39.2538	0.9709	55.3494	0.9983	54.7083	0.998
5	45.5844	0.9906	55.876	0.9984	54.9994	0.9981	63.3388	0.9997	63.3339	0.9997
4	58.5587	0.9991	63.3388	0.9997	63.3388	0.9997	73.6122	1	73.5598	1
3	63.3388	0.9997	73.6122	1	73.6122	1	Inf	1	Inf	1
2	73.6122	1	Inf	1	Inf	1				
1	Inf	1								
8 im8	Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
13	32.0872	0.81	32.811	0.8262	30.1336	0.7448	32.6556	0.8377	31.8809	0.821
12	35.7715	0.9036	35.6789	0.8927	33.1178	0.8377	38.597	0.9484	37.517	0.9364
11	38.5572	0.9402	41.2864	0.9661	38.773	0.9458	39.6918	0.9562	38.9776	0.9509
10	43.4785	0.98	42.2517	0.9705	39.9004	0.9555	42.3981	0.9761	41.1968	0.9698
9	45.0995	0.9849	44.756	0.9835	42.7777	0.9757	43.3407	0.9794	43.0293	0.9783
8	46.4841	0.9887	46.3533	0.9875	43.823	0.9792	45.2193	0.9862	44.8668	0.9852
7	47.2199	0.9894	47.9411	0.9911	46.1979	0.9875	46.3392	0.9888	45.9174	0.9877

6	49.2682	0.9932	48.3774	0.9915	47.555	0.9901	58.6484	0.9991	58.4854	0.999
5	49.8127	0.9937	58.6484	0.999	58.6484	0.9989	63.3388	0.9997	63.3344	0.9997
4	58.6484	0.999	63.3388	0.9997	63.3388	0.9997	73.6122	1	73.5656	1
3	63.3388	0.9997	73.6122	1	73.6122	1	Inf	1	Inf	1
2	73.6122	1	Inf	1	Inf	1				
1	Inf	1								
9. im9	Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
10	33.0717	0.9158	30.1222	0.8565	30.8698	0.8786	31.0437	0.8821	30.1835	0.8684
9	34.1948	0.9291	32.3357	0.9042	32.1794	0.9007	32.702	0.9134	31.7067	0.8996
8	36.4167	0.9567	33.819	0.9243	33.7366	0.9265	33.728	0.9257	32.724	0.914
7	37.9686	0.9671	35.4577	0.9463	34.7298	0.9369	47.3524	0.9936	45.9934	0.9921
6	39.607	0.9773	36.5627	0.9546	47.4178	0.994	58.7376	0.9994	58.2793	0.9994
5	40.9241	0.9819	48.985	0.9957	58.7243	0.9994	73.5946	1	67.0761	1
4	51.519	0.9977	58.7294	0.9994	73.5656	1	Inf	1	Inf	1
3	63.2009	0.9998	73.6122	1	Inf	1				
2	72.3293	1	Inf	1						
1	Inf	1								
10. im10	Block 2		Block 4		Block 6		Block 8		Block 10	
Layer	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
7	30.8429	0.9615	42.6864	0.9956	40.3339	0.9928	39.0565	0.9906	39.0271	0.9906
6	32.4207	0.9722	56.1467	0.9997	56.0308	0.9998	54.1246	0.9996	54.0613	0.9996
5	33.6322	0.9773	73.0682	1	68.0756	1	67.9833	1	68.0155	1

Layer	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
4	45.996	0.998	Inf	1	Inf	1	Inf	1	Inf	1
3	58.7344	0.9999								
2	72.7258	1								
1	Inf	1								

APPENDIX E

BIODATA OF THE AUTHOR

Mohd Shahidan bin Abdullah has born in Kedah, Malaysia in 1960, has obtained a B.Sc (Hons) in Computer Science from Universiti Kebangsaan Malaysia in 1984. He got his Master Degree in Computer (Database) from Faculty of Science and Information Technology at Universiti Putra Malaysia in 1998. Currently he is a senior lecturer at Advance Informatics School in University Technology Malaysia at Kuala Lumpur. Following were the publications published by the author:

1. Shahidan M. Abdullah, Azizah A. Manaf, and Mazdak Zamani. Recursive Reversible Image Watermarking Using Enhancement of Difference Expansion Techniques. Journal of Information Security Research. Volume 1 Number 2. June 2010. Pages 64-70.

2. Shahidan M. Abdullah, Manaf, Azizah A.,"Multiple layer reversible images watermarking using enhancement of difference expansion techniques" , 2nd International Conference on Networked Digital Technologies, NDT July 2010, Prague, Czech republic, Book Chapter, Springer Verlag, Tiergartenstrasse 17, Heidelberg, D-69121, Germany Proceeding Networked Digital Technologies, v 87 CCIS, n PART 1, p 333-342, 2010.

3. Shahidan M. Abdullah, Manaf, Azizah A.; Zamani, Mazdak; "Capacity and quality improvement in reversible image watermarking approach", Proc. of the IEEE Networked Computing and Advanced Information Management (NCM) Seoul, Korea (South, 2010 Sixth International Conference on Publication Year: 2010 , ISBN: 978-1-4244-7671-8 Page(s): 81 – 85.

4. Shahidan M. Abdullah, Manaf, Azizah A.; Akram M. Zeki, "Recursive Reversible Image Watermarking Using Enhancement Difference Expansion Techniques", Proc. of the IEEE International Conference on Computer Applications, Lahore India, ICICCA2010, pg 31-36, 2010.

5. Shaamala Abduljabbar , Shahidan M. Abdullah, Manaf, Azizah A. (2011). Study of the effect DCT and DWT domains on the imperceptibility and robustness of Genetic watermarking. IJCSI International Journal of Computer Science. Vol. 8, Issue 5, No 2, page 220-225

6. Mazdak Zamani, Azizah Abdul Manaf, Rabiah Ahmad, Akram Zeki, and Shahidan M Abdullah. "A Genetic-Algorithm-Based Approach for Audio Steganography". *International Conference on Communities and Communications. World Academy of Science, Engineering and Technology 54 2009*. ISSN: 2070-3740. Pages: 359-363. 24-26 June 2009. Paris, France. (Indexed by ISI; Published by World Academy of Science, Engineering and Technology)

7. Mazdak Zamani, Azizah Abdul Manaf, Rabiah Ahmad, Akram Zeki, and Shahidan M. Abdullah. "Genetic Algorithm as an Approach to Resolve the Problems of Substitution Techniques of Audio Steganography". The 2009 International Conference on Genetic and Evolutionary Methods. 13-16 July 2009. ISBN: 1-60132-106-6 and 1-60132-092-2. Pages 170-175. Las Vegas, Nevada, USA. (Indexed by DBLP, PubZone, BibSonomy, Microsoft Academic Research, CSB; Published by World Academic Union-World Academic Press)