FRACTAL IMAGE COMPRESSION FOR COLORED IMAGES

By

Rawan Zaghloul

Supervisor

Dr. Sami Serhan

This Thesis Was Submitted In Partial Fulfillment Of The Requirements For

The Master's Degree Of Science In Computer Science

Faculty of Graduate Studies

The University of Jordan

August, 2005

Committee Decision

This thesis (Fractal Image Compression for Colored Images) was successfully defended and approved on August 10, 2005.

<u>**Examination Committee**</u>	<u>**Signature**</u>
Dr. Sami Serhan (Chairman) **Associate Prof. Compiler Design**	……………………..
Prof. Nadim Obied (Member) **Prof. Artificial Intelligence**	……………………..
Dr. Mohammed-Al Zubi (Member) **Associate Prof. GIS**	……………………..
Dr. Moussa Habib (Member) **Assist Prof. of Computer Engineering** **(University of Princess Summaya for Technology)**	……………………..

Dedication

Acknowledgement

I have found that it is hard to find words about a teacher who helps me during my work in the thesis. I would like to thank my supervisor Dr. Sami Serhan for supporting me and for making this research project valuable experience for me, also I want to thank him for his advice, assistance, prompting, and encouragement.

I was fortunate enough to have Dr. Moussa Habib as my Image Processing teacher. It was in this course that I learned that Dr. Habib challenges his students to understand the aspects of image processing, and he does so by giving his students as much of his time as he expects them to devote to their studying of image processing, and I want to thank him for his valued assistance and help through out the work in the thesis.

In addition, I want to extend my thanks to my family and my friends for uplifting me when I am down, for pushing me when I wanted to stop, and for teaching me how to take a break and have fun.

Most importantly, I would like to thank the Glorious God for all of the efforts I have put into this thesis. If not for God's awesome creation of the universe, I would not have the zeal for image processing that I have.

List of Contents

List of Tables

List of Figures

List of Abbreviations

IFS:	Iterated Function System
RGB:	Red Green Blue
HSV:	Hue Saturation Value
PIFS:	Partitioned Iterated Function System
MRCM:	Multiple Reduction Copy Machine
RMS:	Root Mean Square
RMSE:	Root Mean Square Error
M:	Magnocellular or large bodied
P:	Parvocellular or small bodied
LGN:	Lateral Geniculate Nuclei
V1:	Visual Area 1
PET:	Position Emission Tomography
MRI:	Magnetic Resonance Imaging
PSNR:	Peak Signal to Noise Ratio

List of Appendices

FRACTAL IMAGE COMPRESSION FOR COLORED IMAGES

By

Rawan Ismail Zaghloul

Supervisor

Dr. Sami Serhan

ABSTRACT

Currently investigations concerning fractal coding schemes exclusively concentrate on grayscale images, while colored images didn't have the same chance of researching as grayscale ones. Though in order to present complete encoding scheme, color information has to be included.

In this thesis, we have proposed six models, they have a significant difference over other models, which is the use of biological model (the Hue Saturation Value model) which acts as the human color sensation. In fact, the first five models try to quantize color planes in different ways, but unfortunately, they didn't achieve the predictable results, since they result a big distortion around edges. So, another model was proposed, to cover the errors of the first five models, which called the *Three-Channel-HSV-Model with Different Fractal*

Encoders, using this model we reach a compression ratio in average 60:1. Then we compare our results with the results of other popular models.

The experimental results of this study offer clear tools and guidelines for the researchers to fractal compress colored images. We recommend using the last model to fractal compress colored images, especially for images which have harmonic variations in colors.

CHAPTER ONE

INTRODUCTION

The purpose of this chapter is to give the reader a fairly clear idea about fractal image compression, which helps the reader to follow the scientific contents of the following chapters. It is written with a concern for simplicity and clarity, while avoiding as much as possible talking in details.

1.1 Background

Every day, new uses for computer graphics are devised. In the past, home computers were used only to manipulate text and numeric information, but nowadays, they have to work with detailed images. The increased use of pictures in computing has led to many advances in hardware and software, and strained computer resources in many ways. One area that is particularly affected is the computer's storage space for graphics, which requires a great deal of it. In 1988, Michael Barnsley proposed fractal image compression as a technique to store images in a small amount of space. His idea was based on his work with fractal mathematics and his observation that most natural scenes contain affine redundancy or subsections that look very similar.

Fractal compression for the image is partitioning the image into image blocks called ranges. Each range is coded by a reference to some other part of the image and by some transformation parameters. These parameters describe how the referenced image part has to be adjusted with respect to contrast and brightness in order to give a good approximation to the range to be encoded. When one uses fixed length codes for the transformation parameters, the size of the fractal code is proportional to the number of blocks of the partitioning image. Therefore, in order to obtain high compression ratios, only a small number of blocks are allowed. Thus, the key point in fractal compression is to partition the image into a small number of blocks that are similar to other image parts under certain transformations.

The main principle of encoding an image using fractals is the observation that self similarity is found within images and is extractable, this employs an affine transform. The affine transform allows all possible two dimensional transformations, including rotation and translation as will as most importantly scaling to be performed. An Iterated Function System (IFS) consists of a set of these transforms, which in some way make up the whole image. The fundamental idea behind a fractal compression system is that if an image can be defined in terms of a self similar set of transforms, and each transform can be described with an affine transform, then a complete description of the image can be achieved within the numbers in the set of affine transforms.

The encoding process of an image involves searching for each range block a domain block that is closest in similarity, and then encoding a relevant transform and location. Image decoding starts with an initial image, possibly just a plain gray image, and the set of transforms are repeatedly applied until the attractor (Decoded image) is closely reached.

Finding the optimal partitions for fractal coding purposes is a hard problem. Many partitioning methods have been proposed for fractal image compression; among the most widely known are, Fisher's quadtree scheme and Fisher's and Menlove's horizontal-vertical partitions. In fractal image compression the encoding step is computationally expensive. A large number of sequential searches through a list of domains are carried out when trying to find a best match for another image portion. Many methods were implemented for accelerating the encoding procedure in fractal image compression for grayscale images. Currently the investigations concerning fractal coding schemes nearly exclusively concentrate on grayscale images, though in order to present complete encoding scheme color information has to be included. Therefore color image coding based on fractal techniques demands further research.

1.2 Historical Perspective

The term fractal is invented in 1982 by Mandelbrot, who often characterized as the father of fractal geometry. A fractal as defined by Mandelbrot is: *"a shape made up of parts similar to the whole in some way"*.

The first suggestion on fractal image compression was in 1988, by Michael Barnsley. He proposed fractal image compression as a technique to store images in a small amount of space.

In 1989, Arnaud Jacquin, - a graduate student of Barnsley- submitted a PhD thesis with a computer implementation of the compression algorithm, this algorithm has been patented by Barnsley, and it is fully described in fractal image compression.

In 1992, several papers and series of lecture notes by Yuval Fisher in the University of California have been published, they contain many improvements, and alternatives to Barnsley's algorithms, and he introduced different methods for image partitioning (Fisher, 1992). Also A.Jacquin proposed a new approach in image coding, based on fractal theory of iterated transformations (Jacquin, 1991).

In 1994, Brend, Paul Mols, and Stephan F.Simon, proposed a method of fractal image compression for Red Green Blue (RGB) images. They proposed the master plane algorithm to be used (Brend et al., 1994).

In 1996, a theory of generalized fractal transform was introduced by Bruno Forte and Edward R.Vercay of the University of *Degli Sttudi di Verona*. It aims to present a unified treatment of the various fractal transform methods, which have been based on the method of IFS (Forte and Vrscay, 1996).

In 1998, Katsuhiko Nakano and Masahiro Nakagawa proposed a new method to increase the efficiency of fractal image encoding for gray images and extend it to be used over the RGB images (Nakano and Nakagawa, 1998). F.L.de Oliveira, V .Mendonca and J. Dias, introduced a new fractal transformation method to improve the quality of fractal encoded images (Oliveira et al., 1998).

In 2003, S.K. Ghosh, Jayanta Mukherjee and P.P.Das, introduced a randomized method for searching fractal image coding scheme, which is able to improve the speed of block oriented fractal image compression (Ghosh et al., 2003).

1.3 Thesis' Objectives and Main Contributions

The general objective of the thesis is to spot the light on fractal compression for colored images, because most of the work was applied on grayscale images, while a relatively little attention has been given to the encoding for colored images.

After reading many researches about this topic, we become more interested and involved with the idea of biological sensation of the color. This idea represents our motive to reach a new vision for color image compression based on the biological model of color

- the Hue Saturation Value (HSV) model. Then to evaluate our work, we have to compare our results with previous results of other popular models.

In this thesis, we have proposed six models; the first five models didn’t achieve good results, because of the use of quantization before fractal compressing the image. A compression ratio in average 60:1 was achieved by applying the sixth model. Later a comparison between our results and the results of previous models was done.

1.4 Thesis Organization

The thesis has covered the aspects of the following chapters:

- *Chapter Two*, "Theory of Fractals and Iterated Function Systems": This chapter introduces some of the basic concepts of fractals and IFSs.
- *Chapter Three*, "Fundamental Principles of Colors": this chapter explains several principles of colors, discusses the definition of color and the mechanics of human visual system, and defines the RGB and HSV color models.
- *Chapter Four*, "Fractal Image Compression for Gray Scale Images": This chapter introduces Jacquin’s Partitioned IFS (PIFS) scheme. Then it summarizes the theoretical basis of a block oriented fractal image encoding scheme for gray scale images.
- *Chapter five*, "Fractal Color Image Compression": This chapter discusses the recent algorithms that are used in fractal color image compression, and then, describes the components of the proposed compression system.

- *Chapter Six*, "Fractal Image Compression for Colored Images using Biological Model": In this chapter we will look at the proposed models, which are about compressing colored images using fractal encoding technique.
- *Chapter Seven*, "Three-Channel-HSV-Model Results and Analysis": this chapter illustrates the experimental results of the sixth proposed model.
- *Chapter eight*, "Conclusions and Future Work".
- *Appendix-A*, "Results of Applying the Three-Channel-HSV-Model Part One": Results of applying the Three-Channel-HSV-Model over nine testing images.
- *Appendix-B*, "Results of Applying the Three-Channel-HSV-Model Part Two": Additional results for applying the Three-Channel-HSV-Model over 20 testing images.
- *Appendix-C*, "Sample Code".

CHAPTER TWO

THEORY OF FRACTALS AND ITERATED FUNCTION SYSTEMS

This chapter summarizes some of the basic concepts of fractals and IFSs, their properties, characteristics, and a brief description of their mathematical notations is also included.

2.1 Basic Concepts

Imagine a special type of copy machine that reduces the size of the image by a factor of (1/2) and replicates it three times as shown in figure 2.1.

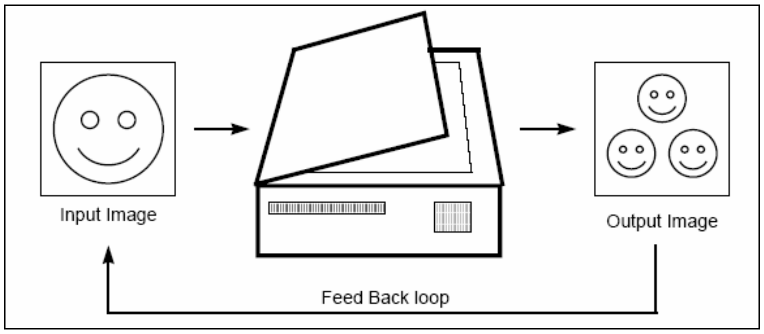


Figure 2.1: A copy machine that makes three reduced copies of the input image. (Fisher, 1992)

Now, let us do the following; run the machine several times over several input images, what happens? You can see that images will converge to the same final image, this is what we called the *attractor* of the image, which is the sierpinski triangle in the case shown in figure 2.2.

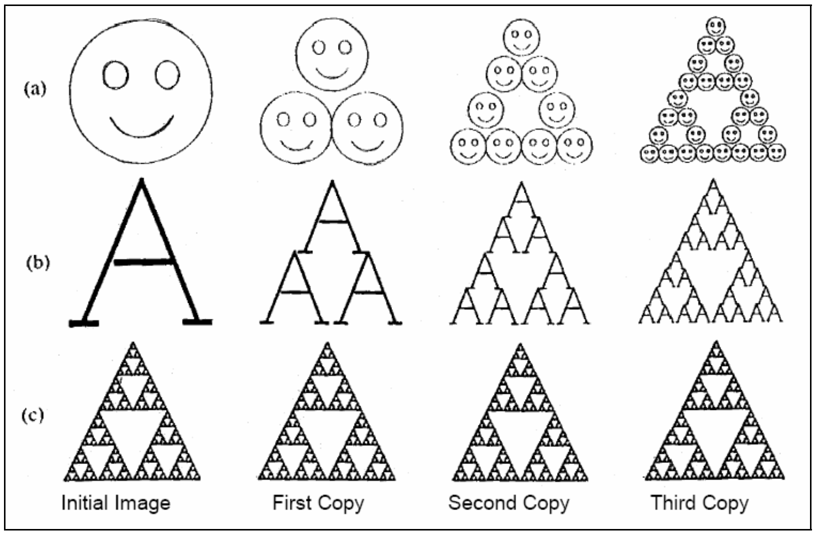


Figure 2.2: The first three copies generated on the copying machine of Figure 2.1. (Fisher, 1992)

Since the copying machine reduces the input image, any initial image placed on the copying machine will be reduced to a point as we repeatedly run the machine. In fact the final image is determined by the transformations of the input image when running the Multiple Reduction Copy Machine (MRCM) in a feedback loop. These transformations must satisfy the contraction mapping principle as explained later in this chapter. (Texas Instruments Europe, 1997)

2.2 Properties of Fractals

1- The word fractal comes from the characteristic of a fractal that it has a fractional dimension. The fractal dimension may not be an integer, it may have fractional values. (Peitgen et al., 1992), (Hearn, Baker, 1997)

2 - The property of self-similarity is one of the central concepts of fractal geometry, it is the property by which magnified subsets appear similar or identical to the whole and to each other. Thus fractal shapes are self-similar and independent of scale.

2.3 Metric Spaces

A metric space is a set of points along with a function that takes two elements of the set and gives the distance between them. The set can be a set of points or a set of images.

Definition A *metric space* (X, d) is a set X together with a real-valued function $d : \mathrm{X} \times \mathrm{X} \to R$, which measures the distance between pairs of points x and y in X. d is called a *metric* on the space X when it has the following properties (Texas Instruments Europe, 1997):

1- $d(x, y) = d(y, x), \quad \forall x, y \in \mathrm{X}$.

2- $d(x, y) \geq 0, \quad \forall x, y \in \mathrm{X}$.

3- $d(x, y) = 0 \;\; iff \;\; x = y, \;\; \forall x, y \in \mathrm{X}$.

4- $d(x, y) \leq d(x, z) + d(z, y), \;\; \forall x, y, z \in \mathrm{X}$ (Triangle inequality).

This means, any distance function that satisfies the conditions from 1 – 4 is called a metric, and a set of points X together with a metric d defined on X is called a metric space, and is denoted (X,d). Some examples of metrics are:

1- The standard Euclidean metric, which finds the distance between two points (x_1,y_1) and (x_2,y_2) in the plane. Its function as shown in equation 2.1

$$d(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}, \quad \forall x, y \in \mathrm{R}^2 \qquad \text{.........} \ (2.1)$$

2- Another metric between the points (x_1,y_1) and (x_2,y_2) is

$$\max\{|x_{1\_}x_2|, |y_1 - y_2|\} \qquad \text{.........} \ (2.2)$$

3- The root mean square (rms) metric is one of the most popular metrics that is used to define the distance between two images, as shown in equation 2.3.

$$d(f,g)=\sqrt{\sum(f(x,y)-g(x,y))^2}\,, \qquad \text{……… (2.3)}$$

where *f* and *g* are two distinct images. (Fisher, 1994)

2.4 Affine Transformations

Affine Transformations are determined by combination of rotations, scalings and translations of the coordinate axes.

2.4.1 The General form of an affine transformation

The General form of an affine transformation is given in equation 2.4.

$$W\begin{pmatrix} x \\ y \end{pmatrix}=\begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix}+\begin{pmatrix} e \\ f \end{pmatrix}=\begin{pmatrix} ax+by+e \\ cx+dy+f \end{pmatrix}=AX+T, \qquad \text{……… (2.4)}$$

where A is a 2×2 real matrix and $T=\begin{pmatrix} e \\ f \end{pmatrix}$ represents translations.

If the translations, rotations, and scalings that generate *W* are known in advance, then we can represent the vector *A* in its polar form as depicted in equation 2.5.

$$A=\begin{pmatrix} a & b \\ c & d \end{pmatrix}=\begin{pmatrix} r\cos\theta & -s\sin\phi \\ r\sin\theta & s\cos\phi \end{pmatrix}, \qquad \text{……… (2.5)}$$

where *r* is the scaling factor on *x*, *s* is the scaling factor on *y*, θ is the angle of rotation on *x*, ϕ is the angle of rotation on *y*, *e* is the translation on *x* and *f* is the translation on *y*. (Ali and Clarkson, 1991)

2.4.2 Contractive Transformations

A transformation *W* is said to be contractive if for any two points P_1 and P_2 the distance $d(W(P1),W(P2))<sd(P1,P2)$, where $0\le s<1$ and d is the distance. This formula says the application of a contractive map always brings points closer together by some factor *s* less than 1. (W.L.NG and Cohen, 1994)

2.4.3 The Contractive Mapping Fixed Point Theorem

This theorem says something that is intuitively obvious: if a transformation is contractive then when applied repeatedly starting with any initial point, we converge to a unique fixed point. If X is a complete metric space and $W : X \to X$ is contractive, then W has a unique fixed point. (W.L.NG and Cohen, 1994), (Wohlberg and Jager, 1999)

2.5 What is IFS?

An *Iterated Function System* (IFS) is a collection of contractive affine transformations which express relations between parts of an image. The relations are able to define very complicated details of a picture. An example of an IFS code for the generation of a Sierpinski triangle is given below, consisting of three affine transformations, in matrix form:

Table 2.1: Affine transformations for a Seirpinski triangle

W	***A***	***B***	***C***	***D***	***e***	***f***
W1	0.50	0	0	0.5	0	0
W2	0.5	0	0	0.5	0.5	0
W3	0.5	0	0	0.5	0.25	0.5

Any MRCM - as a type of IFSs - is described by a set of affine transformations w_1, w_2,…, w_n. For any initial image A small affine copies $w_1(A)$, $w_2(A)$,…, $w_n(A)$ are produced. Finally, the machines overlays all these copies into one new image, the output $W(A)$ of the machine:

$$W(A) = w_1 \cup w_2 \cup w_3 \cup \ldots \cup w_N(A) \qquad \ldots\ldots\ldots (2.6)$$

As illustrated in figure 2.3, the result of running the MRCM in a feedback mode - thus corresponds to iterating the operator *W*, will converge toward a fixed image or attractor as shown in the 9th and 10th iterations of generating the Seirpinski triangle.

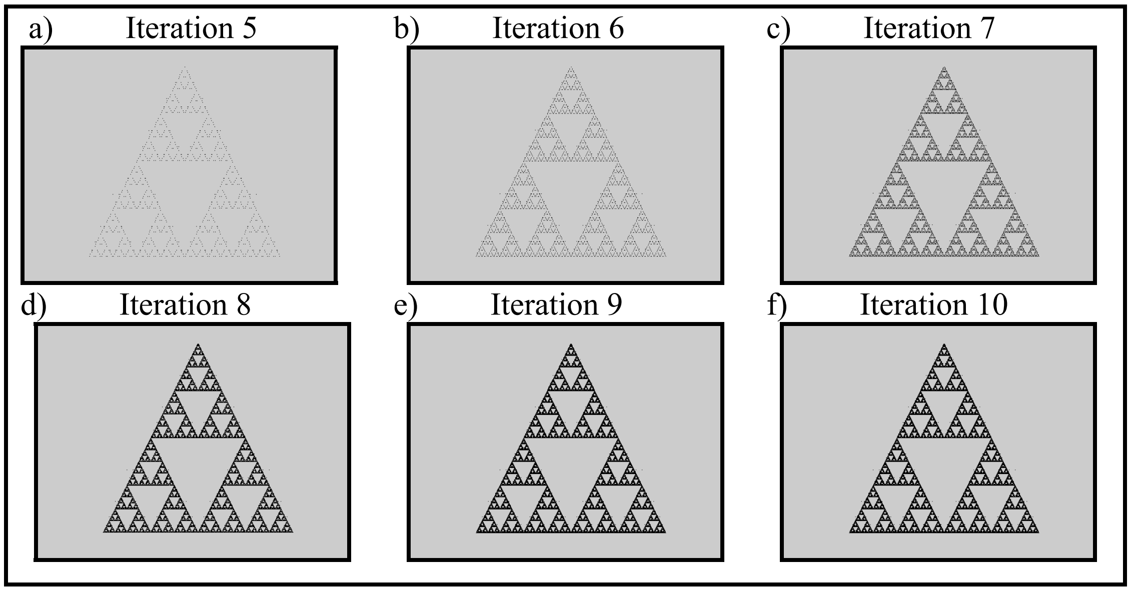


Figure 2.3: Generating Seirpinski triangle, figures from a-f represent its iterations from 5 to 10.

As a result, starting with some initial image A_0, we obtain $A_1 = W(A_0)$, $A_2 = W(A_1)$ and so on, figure 2.4 summarizes this process. (Kominek, 1997)

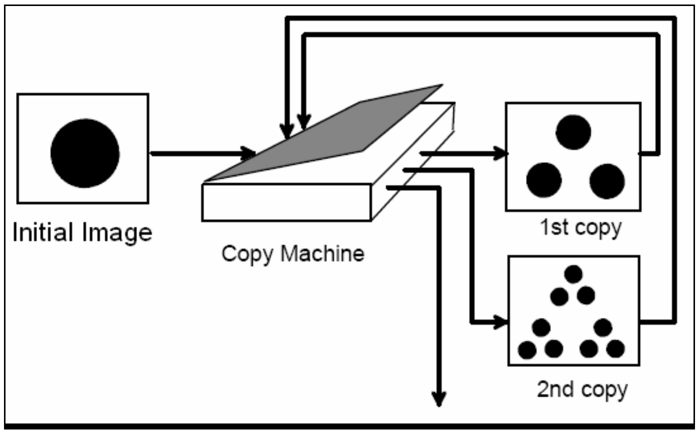


Figure 2.4: Diagram of a MRCM

CHAPTER THREE

FUNDAMENTAL PRINCIPLES OF COLORS

The purpose of this chapter is to introduce several principles of colors. At the beginning, section 3.1 discusses the definition of color, section 3.2 briefly summarizes the mechanics of human visual system and section 3.3 deals with color spaces, especially the RGB and HSV color models and there transformations from one to another.

3.1 What is Color?

Everything appears colored. For example, one can not look at grass without seeing green, or look at the sky without seeing blue. The effect of the color on human senses makes it seem as if it is an aspect of reality, but of course it is not. Neither the light reflected from grass nor the light scattered from the sky, has color; both are just collections of photons. Colors are created in our minds as a response to the light. So, we can define the color as how a human observer interprets light. Color can not be defined in physical terms. Since it is not a physical property, but once a behavioral definition is in place, it can be used to understand the correlations between color sensation and the physical properties of the light, as an example, take a colored square then place it under a variable intensity light. When the intensity of the light is varied, the color of the square stays the same. Color must therefore be something which is invariant with respect to light intensity. (Sangwine and Horne, 1998)

In fact, although the process followed by the human brain in perceiving and interpreting color is a physiopsychological phenomenon that is not yet fully understood, the physical nature of color can be expressed on a formal basis supported by experimental and theoretical results.

Basically, the colors that humans perceive in an object depend on the nature of the light reflected from the object. As shown in figure 3.1, visible light is composed of a relatively narrow band of frequencies in the electromagnetic spectrum. A body that reflects light that is balanced in all visible wavelengths appears white to the observer. However a body that favors reflectance in a limited range of the visible spctrum exhibits some shades of color. For example, green objects reflect light with wavelengths primarily in the 500 to 570 nm range, while absorbing most of the energy at other wavelengths. (Gonzalez, 2001)

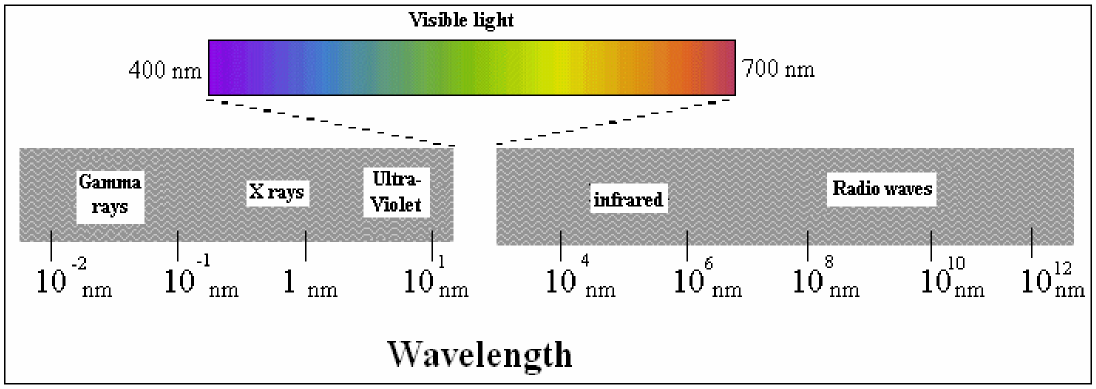


Figure 3.1: Electromagnetic spectrum.

3.2 The Visual Pathway

Color perception occurs in stages. Each stage has a different location in the brain. The pipeline of stages is called the visual pathway as represented in figure 3.2, it includes three parts: the retina, the lateral geniculate nucleus, and the cortex.

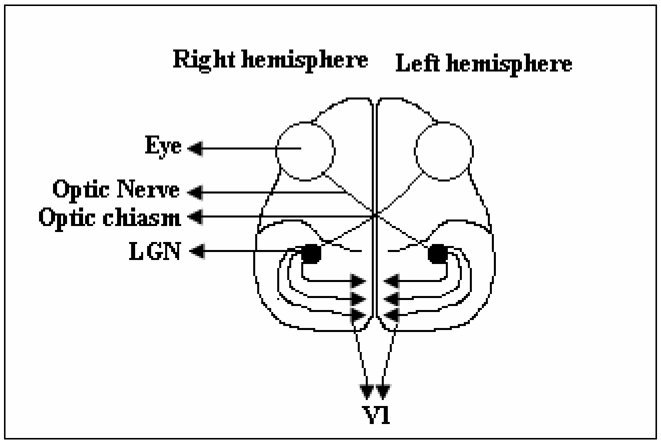


Figure 3.2: The visual pathway in humans

The first stage in the visual pathway is the retina. Light is focused by the cornea and lens onto the retina*. The retina is composed of a number of layers. Specialized cells in one layer of the retina called photoreceptors, they absorb the light and convert it in to a neural signal. These neural signals are collected and further processed by cells in other layers of the retina before being passed to the brain along the optic nerve.

The photoreceptors lie in the outer layer of the retina, furthest from the light. The other cells in the retina lie in layers on top of them, partially obscuring that light, although in the fovea, where visual acuity is greatest. The optic nerve connects to the cells in the innermost layer, called the retinal ganglion cells, so it must punch through the outer

* Retina: is a thin layer of tissue at the back of the eye.

(photoreceptor) layer to escape the eye; and this leaves a small spot on the eye which has no photoreceptors – the blind spot. The cells in the inner retinal layer can be divided into two classes, called M (for Magnocellular, or large bodied) and P (for Parvocellular, or small bodied) cells. Only the P cells carry a color signal. The optic nerve passes from the eye to a pair of small midbrain structures called the Lateral Geniculate Nuclei (LGN). These don't seem to process the nerve signals in any significant way. From the LGN, the visual signal passes to V1 (visual area 1, also called area 17) which lies in the cleft between the two brain hemispheres at the back of the brain. V1 is laid out like a map of the retina: each point in V1 processes signals from a corresponding point on the retina. Each hemisphere of the brain carries half of V1. The left hemisphere holds the part of V1 that deals with the right visual field and the right hemisphere holds the part that deals with the left. Within V1, there are subgroups of cells that are color sensitive. These cells are distributed throughout V1 in clumps, called "blobs" and receive input from the P cells (Livingstone and Hubel, 1984). From V1, the color signal from the blobs passes to other visual areas, imaginatively named V2, V3 and V4, from these, V4 is the most crucial. It is clearly identified as being color specific in Position Emission Tomography (PET) and functional Magnetic Resonance Imaging (MRI) images of the human brain, and the damage in this area reduces or eliminates the ability to see color. However, V4 seems to be involved in other kinds of visual processing. The color signal doesn't pass to some areas of the brain, most notably V5, which deals with the detection of motion.

Color perception begins with the light absorption by the photoreceptors in the retina. There are two main kinds of photoreceptors, called rods and cones. Rods, which are long and thin, function only in darkness and do not contribute to color perception. Cones, which have a tapered shape, do not respond very well in dim lighting; their purpose is to handle vision during day light. Color perception is best on the activity of the cones. There are three different types of cones, called L, M and S, which stand for Long, Medium and Short wavelengths. Each cone type absorbs light over a broad range of wavelengths, but has its peak absorption at a different wavelength. The L cones peak at 570 nm, the M cones at 545 nm and the S cones at 440 nm, as depicted in figure 3.3. Cone absorption is minimal outside the range (400-700) nm, which therefore defines the visible spectrum. The cones are occasionally called R, G and B, instead of L, M and S, but the red sensation is not due to activity in the R cone but to a comparison of R and G (L and M) cones. (Sangwine and Horne, 1998)

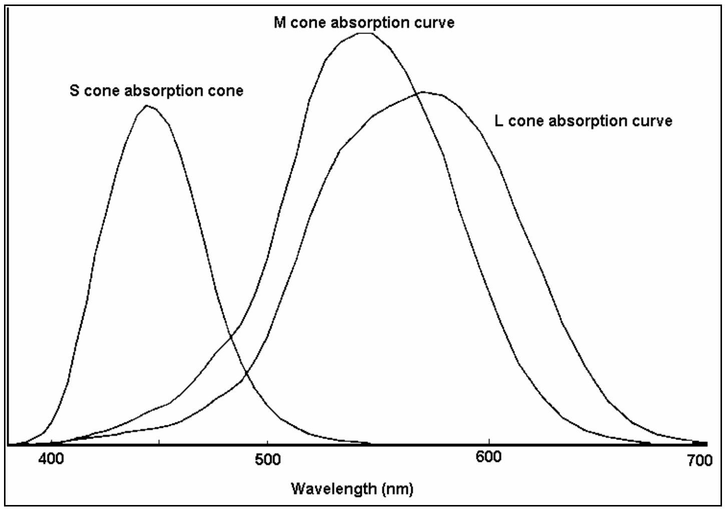


Figure 3.3: Absorption curves for the L, M and S cones.

3.3 Color Spaces

Color spaces are three dimensional arrangements of color sensations. Colors are specified by points in these spaces. Color spaces which are presented here are only the

RGB and the HSV, because they are the only color spaces that we have used through out the thesis.

Most color monitors on the market today are based on the RGB color space. Each pixel on the screen is made up of three phosphors (colors): Red, Green and Blue. When all of them are illuminated, the pixel appears white, otherwise the pixel appears black. Various combinations of the three phosphors result in a large number of distinct colors and shades.

Unlike the RGB color space, which is hardware oriented, the HSV color space is user oriented, based on the more intuitive appeal of combining hue, saturation and value elements to create a color. (Sangwine and Horne, 1998)

3.3.1 The RGB Color Space

The RGB space is the most frequently used color space in image processing. Since color cameras, scanners and monitors are most often provided with direct RGB signal input or output, this color space is the basic one, which can be transformed into other color spaces.

Each color, which is described by its RGB components, is represented as a point, and can be found either on a surface or inside a cube as shown in figure 3.4. All gray colors are placed on the main diagonal of this cube from black (*R*=*G*=*B*=*0*) to white (*R*=*G*=*B*=*maximum value*).

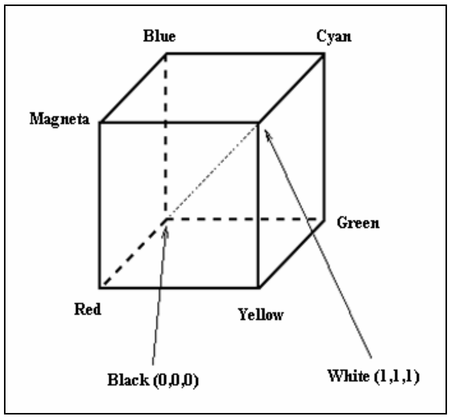


Figure 3.4: The RGB cube.

The main disadvantage of the RGB (as known from the theory of color spaces) in the applications involving natural images is a high correlation between its components, this makes the RGB space unsuitable for conventional compression techniques. But in fact, this point was the essential unit of utilization used by Brend, Paul and Stephan in their work in fractal image compression for RGB images, as explained in chapter 5. Other disadvantage of the RGB space is its psychological non-intuitivity, especially when it is compared with other spaces such as the HSV color space. (Gonzalez, 2001)

3.3.2 The HSV Color Space

The HSV color space was created in 1978 by Alvy Ray Smith. It is a nonlinear transformation of the RGB color space. As presented in figure 3.5, the HSV defines a color space in terms of three components: The hue, which represents the impression related to the dominant wavelength of the color stimulus. The saturation, which corresponds to relative color purity (lack of white in color), for example, the saturation for a pure color is equal to 100%. The third component is the value, which represents the gray scaled image.

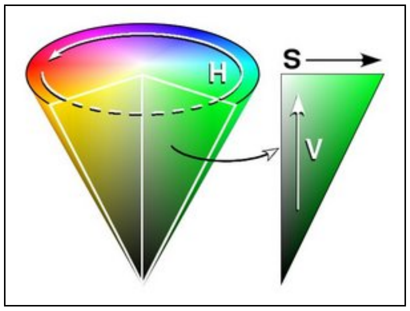


Figure 3.5: HSV color space as a color wheel.

The key idea of using the HSV space in image processing applications is the separation between the gray scaled image (V) from other color components given by H and S. This feature is very useful especially in image filtering for colored images, so, instead of filtering three channels in RGB, only one feature (V) may be used. Other important advantage of using HSV space over other color spaces is its compatibility with human intuition. (Sangwine and Horne, 1998), (Gonzalez, 2001)

3.3.3 Transformation from RGB to HSV

Given a color defined by (R,G,B) where R, G and B are between 0.0 and 1.0, with 0.0 being the least amount and 1.0 being the greatest amount of that color, an equivalent (H,S,V) color can be determined by a series of equations (the set of equations are presented in an algorithmic way), as shown in equations 3.1, 3.2 and 3.3.

Let *MAX* equal the maximum of the (R,G,B) values, and *MIN* equal the minimum of those values. The algorithm can then be written as taken from (Wikipedia, 2001):

$$H = \begin{cases} \left(0 + \frac{G - B}{MAX - MIN}\right) \times 60, if\ R = MAX \\ \left(2 + \frac{B - R}{MAX - MIN}\right) \times 60, if\ G = MAX \\ \left(4 + \frac{R - G}{MAX - MIN}\right) \times 60, if\ B = MAX \end{cases} \qquad \text{............ (3.1)}$$

$$S = \frac{MAX - MIN}{MAX} \qquad \text{............ (3.2)}$$

$$V = MAX \qquad \text{............ (3.3)}$$

The resulting values are in (H,S,V) form, where H varies from 0.0 to 360.0, correspond to the angle in degrees around the color circle where the hue is located. S and V vary from 0.0 to 1.0, with 0.0 being the least amount and 1.0 being the greatest amount of saturation or value, respectively. As an angular coordinate, H can wrap around from 360 back to 0, so any value of H outside of the 0.0 to 360.0 range can be mapped onto that range by dividing H by 360.0 and finding the remainder (as modular arithmetic). Thus, for example, 480 is equivalent to 120. Also note that, the formulas given reflect some other properties of HSV:

- If $MAX = MIN$ (i.e. S = 0), then H is undefined. If S = 0 then the color lies along the central line of grays, so naturally it has no hue, and the angular coordinate is meaningless.
- If $MAX = 0$ (i.e. if V = 0), then S is undefined. If V = 0, then the color is pure black, so naturally it has neither hue, nor saturation.

3.3.4 Transformation from HSV to RGB

Given a color defined by (H,S,V) values, with H, varying between 0.0 and 360.0, indicating the angle in degrees, and with S and V, varying between 0.0 and 1.0, representing the saturation and value, respectively. A corresponding (R,G,B) color can be determined through a series of equations from 3.4-3.18.

First, if S is equal to 0.0, then the resulting color is achromatic, or gray. In this special case, R, G and B are simply equal to V, and H is irrelevant in this situation. When S is non-zero, the following algorithm can be used: (Wikipedia, 2001)

$$H_i = \left\lfloor \frac{H}{60} \right\rfloor \qquad \text{............ (3.4)}$$

$$f = \frac{H}{60} - H_i \qquad \text{............ (3.5)}$$

$$p = (1 - S) \qquad \text{............ (3.6)}$$

$$q = (1 - f\,S) \qquad \text{............ (3.7)}$$

$$t = (1 - (1 - f)S) \qquad \text{............ (3.8)}$$

$$if\ H_i = 0 \rightarrow r = u, g = t, b = p \qquad \text{............ (3.9)}^*$$

$$if\ H_i = 1 \rightarrow r = q, g = u, b = p \qquad \text{............ (3.10)}$$

$$if\ H_i = 2 \rightarrow r = p, g = u, b = t \qquad \text{............ (3.11)}$$

$$if\ H_i = 3 \rightarrow r = p, g = q, b = u \qquad \text{............ (3.12)}$$

$$if\ H_i = 4 \rightarrow r = t, g = p, b = u \qquad \text{............ (3.13)}$$

* *u* in equations (3.9-3.14) represents the unit vector.

$$if\ H_i = 5 \rightarrow r = u, g = p, b = q \qquad \text{............ (3.14)}$$

$$v = \frac{V}{MAX(R,G,B)} \qquad \text{............ (3.15)}$$

$$R = r \times v \qquad \text{............ (3.16)}$$

$$G = g \times v \qquad \text{............ (3.17)}$$

$$B = b \times v \qquad \text{............ (3.18)}$$

CHAPTER FOUR

FRACTAL IMAGE COMPRESSION FOR GRAY SCALE IMAGES

This chapter introduces Jacquin’s Partitioned IFS (PIFS) scheme, which enables one to create a set of transformations that obey the general contractivity requirements. Then it summarizes the theoretical basis of a block oriented fractal image encoding scheme for gray scale images.

4.1 PIFS

Barnsley, was the first to take a step toward solving the inverse problem of finding a suitable IFS code, whose fractal image is to represent the real image (Welsted, 1999).

Deterministic images such as sirpenski triangle can be produced easily with simple IFS as illustrated in chapter 2, in which the entered image is constructed by smaller copies of itself. Applying zooms over this type of images will display the same level of details regardless of the resolution scale. Moreover this type of images is binary i.e. (each pixel represented by 0 or 1). On the other hand, real world images don’t exhibit this type of global self similarity presented in the IFS images. Also, each pixel in real world images belongs to a range of values (gray scale) or vector of values (color). So, if we want to represent this type of images as the attractor of an IFS, then we will need a more general system than the IFS (Welsted, 1999).

He was one of Barnsley's Ph.D. students, called Arnaud Jacquin, who defined a more flexible transform with PIFS, which solved some of the limitations of the standard

IFS. A PIFS relates one area of an image to another which is similar. Jacquin defined large areas called *domain blocks* that are associated with smaller areas called *range blocks*. Each range block is defined as being similar with respect to an affine transform with its associated domain block (Turner, 1998).

Now, let us review how the PIFS machine works. When a partition from the original image called domain, denoted by D_i, is copied (with a brightness and contrast transformation, denoted by w_i) to a part of the produced copy called range, denoted by R_i, this means that, each partition D_i is associated with its transformation w_i implicitly to produce $R_{i.}$ So, given an image *f*, a single copying step in a machine with *N* copies can be written as $W(f) = w_1(f) \cup w_2(f) \cup ... \cup w_N(f)$. This machine run in a feedback loop (its output is fed back as its new input again and again), note that, this is the same notation used in IFS, but here we have an implicit difference, since we say that each partition D_i is associated with its transformation w_{i} **implicitly**.

Therefore, we can conclude that the central improvement in PIFS is: its ability to divide the image into partitions which are each transformed separately. So, the goal of PIFS is to find maps $w_{1,}w_2,\ldots,\ w_3$, collect them into transformation $W = \bigcup_{i=1}^{n} w_i$, show that under certain conditions *W* is contractive, deduce that it has a fixed point and attempt to select *W* so that its attractor is close to some given image.(Fisher, 1994)

4.2 Affine transformations for gray scale images

Here we want to explain the representation of the affine transformations for gray scale images, as used through out the thesis. There are two spatial dimensions denoted by (*x*,*y*) and the gray level denoted by *z*, this is shown in equation 4.1.

$$w_i\begin{bmatrix}x\\y\\z\end{bmatrix}=\begin{bmatrix}a_i & b_i & 0\\c_i & d_i & 0\\0 & 0 & s_i\end{bmatrix}\begin{bmatrix}x\\y\\z\end{bmatrix}+\begin{bmatrix}e_i\\f_i\\o_i\end{bmatrix}, \qquad \ldots\ldots\ldots\ldots (4.1)$$

Where, s_i and o_i control the contrast and brightness of the transformation respectively. It is always implicit, and important to remember, that each w_i is restricted to Di and w_i(Di) = Ri. Since we want *W*(*f*) to be an image, we must insist that $\cup R_i =$ Original Image and that $R_i \cap R_j = \phi$ when $i \neq j$. In the MRCM, this is equivalent to say that the copies cover the whole square page and that they are adjacent but not overlapping.

Running the MRCM in a loop means, iterating the map *W*. We begin with an initial image f_0 and then iterate $f_1 = W(f_0),\ f_2 = W(f_1) = W(W(f_0))$, and so on. We denote the n-th iterate by $f_n = W^{on}(f_0)$. (Fisher, 1994)

4.3 Fixed Points for PIFS

In the PIFS case, a fixed point, or attractor is an image *f* that satisfies $W(f) = f$, that is; when we apply the transformations to the image, we get back the original image. The contractive mapping theorem says that the fixed point of *W* will be the image we got, when we compute the sequence $W(f_0), W(W(f_1)), W(W(W(f_0))), \ldots$, where f_0 is any initial image. So if we have *W* that is contractive in the space of all images, then it will have a unique fixed point that will be an image. (Fisher, 1994)

4.4 The Metric Space

If we want to know when *W* is contractive, we will have to define a distance between two images. There are many metrics to choose from, as illustrated in chapter2. Through out the thesis, we prefer to use the mean square as our metric space. Given image *f* and image *g*, then mean square is as given in equation 4.2.

$$d_{rms}(f,g)=\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i-y_i)^2} \qquad \text{...........} \ (4.2)$$

We use d_{rms} to test the contractivity condition. In the case of gray scale images we need to minimize the error between the transformed domain block given by $[s_iD_i+o_i]$ and the corresponding range block R_i, this is achieved by minimizing the values of s_i and o_i for the transformed block. (Puram, 1999).

The minimum d_{rms} occurs when the partial derivatives with respect to s and o are zero, which occurs when

$$s=\frac{\left[n^2\left(\sum_{i=1}^{n}D_iR_i\right)-\left(\sum_{i=1}^{n}D_i\right)\left(\sum_{i=1}^{n}R_i\right)\right]}{\left[n^2\sum_{i=1}^{n}D_i^2-\left(\sum_{i=1}^{n}D_i\right)^2\right]} \qquad \text{...........} \ (4.3)$$

$$o=\frac{\left[\sum_{i=1}^{n}R_i-s\sum_{i=1}^{n}D_i\right]}{n^2} \qquad \text{...........} \ (4.4)$$

4.5 Construction of Block Oriented Fractal Codes (Fixed Block Codes)

The first decision to be made when designing a fractal coding scheme is in the choice of the type of image partition used for the range blocks. A wide variety of partitions have been used, such as the fixed size square blocks[*], Quadtree, Horizontal Vertical Partitioning and others.

The definition of fixed blocks transformations (the representation of the transformations when the fixed blocks partitioning is chosen) as taken from (Jacquin, 1991) is

$$\forall \mu \in \mathrm{M},\ W(\mu) = \sum_{0 \le i \le N} (W\mu) \,|\, \mathrm{R}_i = \sum_{i \le i \le N} w_i(\mu \,|\, \mathrm{D}_i), \qquad \ldots\ldots\ldots\ldots (4.5)$$

Where, R_i denote non overlapping partition of the image into N range cells, $\mu\ |R_i$ denote the restriction on the image to the cell R_i and w_i denotes an elementary block transformation form the domain cell D_i to the range cell R_i, $w_i = T_i \text{ o } S_i$, where T_i is the geometric part and S_i is the massic part.

The geometric part of the transformation corresponds to map by position and size parameters from the domain cell D_i to the range cell R_i. The most popularly used geometric part is:

$$sizeof(D_i) = 2 * sizeof(R_i) \qquad \ldots\ldots\ldots\ldots (4.6)$$

So, the pixel values of the range block are the average values of four pixels in the domain block. On the other hand, the massic part of the transformation consists in a modification of the pixel values inside the block. It allows changing the gray level information in order to get good approximation of the block R_i. Massic part controls eight possible shuffles of pixels (4 rotations and 4 reflections called isometries), as shown in figure 4.1, a contrast scaling s_i and brightness shift o_i. The massic part of the transformation can therefore be expressed as shown in equation 4.7

[*] The fixed blocks partitioning is used through out the work in this thesis.

$$\mathrm{Si}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_i & b_i & 0 \\ c_i & d_i & 0 \\ 0 & 0 & s_i \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ o_i \end{bmatrix}, \qquad \text{………… (4.7)}$$

Where, a_i, b_i, c_i, d_i correspond to one of the eight isometries, s_i represents a contrast scaling and o_i represents a brightness shift.

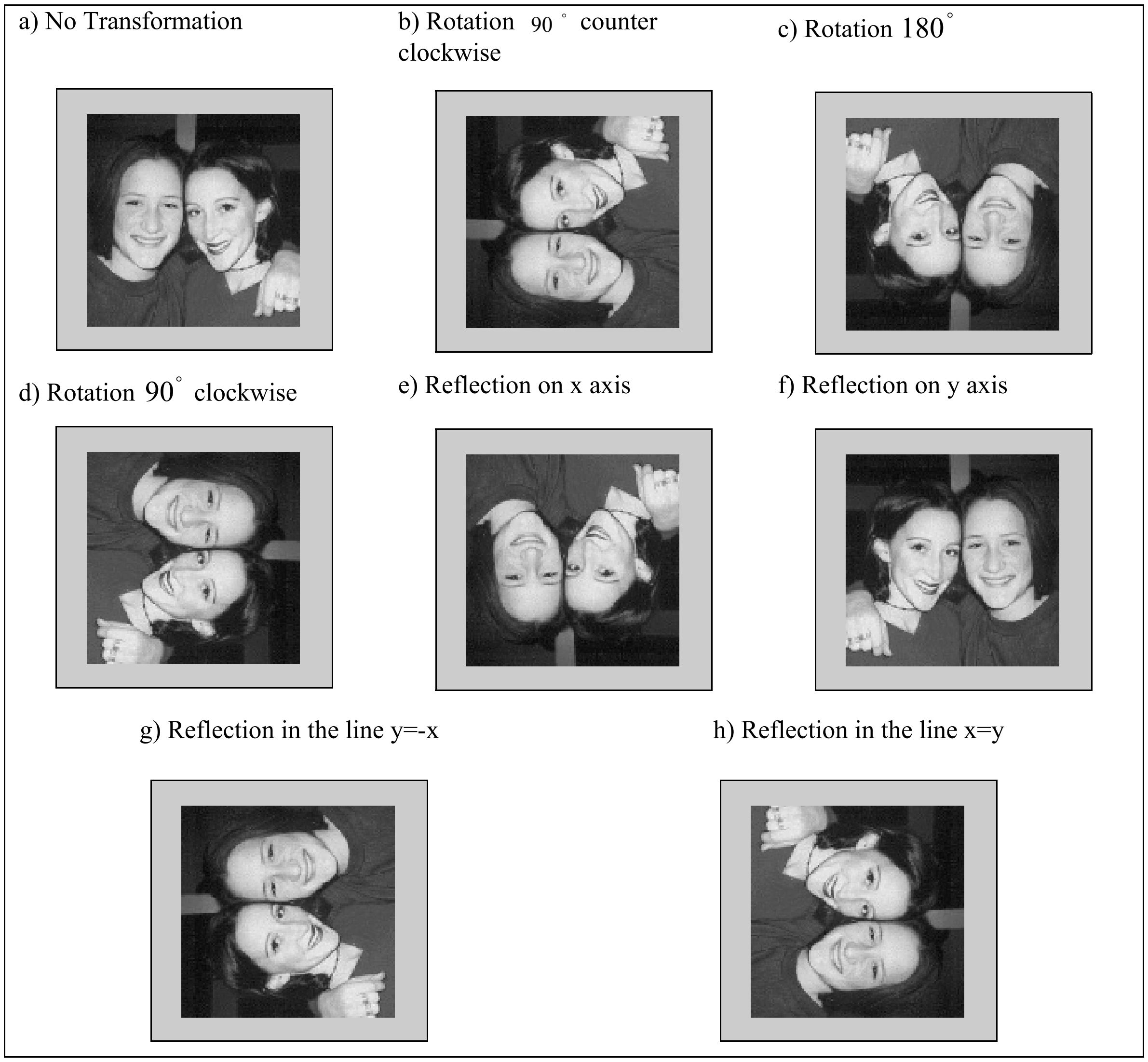

a) No Transformation
b) Rotation 90° counter clockwise
c) Rotation 180°
d) Rotation 90° clockwise
e) Reflection on x axis
f) Reflection on y axis
g) Reflection in the line y=-x
h) Reflection in the line x=y

Figure 4.1: Affine Transformations

In summery, to fractal encode a 256×256 8-bit image using 4×4 range blocks, we need 6 bits for the *x* position, 6 bits for *y* position, 3 bits for the type of the selected affine transformation *w*, 2 bits for *s* and 5 bits for *o*. So, storing the fractal code needs approximately 3 bytes.

4.6 Fractal image compression (General Method)

In the following two sections, we describe fractal image compression (encoding and decoding) using block oriented image partitioning method.

4.6.1 Image Encoding

1. Divide the image into non overlapping blocks (Range Regions).
2. Choose a set of all allowable regions as Domain regions (overlapping blocks).
3. Output the domain partition information (position, scale, rotation) to the output file.
4. Choose a class of affine transformations (isometries) that will be considered when searching for the best domain for each Range Block.
5. While there are uncovered ranges in the list do:
    5.1 Get next range
    5.2 Compare the image data in this range with the transformed data for each possible domain using each possible affine transformation.
    5.3 Output the best affine coefficients obtained for the range to the output file.

4.6.2 Image decoding

1. Read domain partition information and affine transformation from the output file.
2. Create memory buffers for the domain and range screens.
3. Initialize the domain screen buffer to an initial stage using the domain partition information that's read in step 1.
4. While there are uncovered ranges do:

   4.1 Point to next range.

   4.2 Replace this range with the transformed data from the appropriate domain using the affine coefficients stored for this domain.

   4.3 Mark the range as covered. If more iterations are required, copy the contents of the range screens to the domain screen, mark all ranges as uncovered and execute the above loop.
5. Output the final range screen.

Figure 4.2 shows the obtained results when applying Fixed Blocks fractal compression method to the sisters' image. (Mackinnon, 2004), (Fisher, 1992)

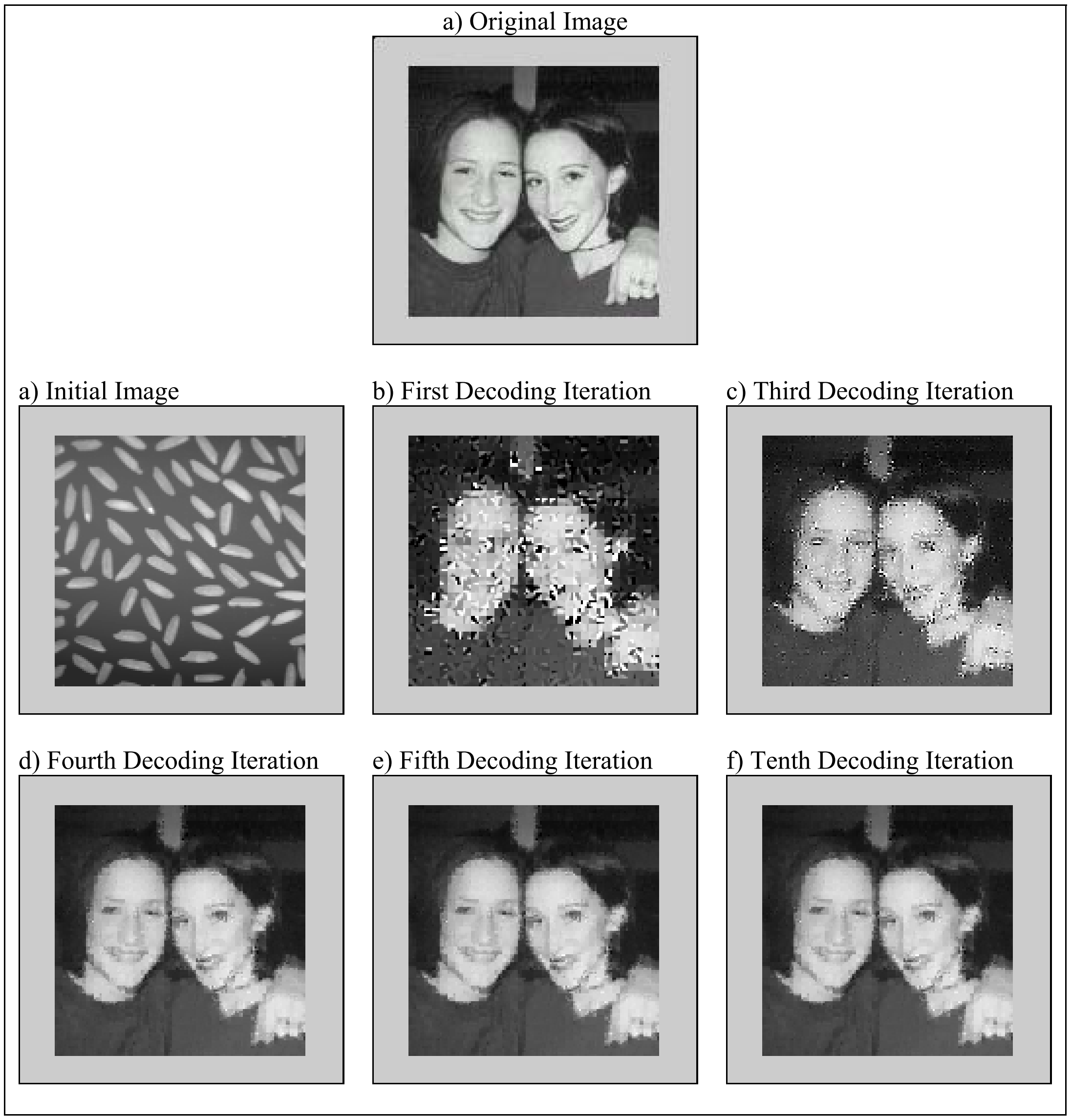

a) Original Image
a) Initial Image
b) First Decoding Iteration
c) Third Decoding Iteration
d) Fourth Decoding Iteration
e) Fifth Decoding Iteration
f) Tenth Decoding Iteration

Figure 4.2: The Fractal Decoding to the Sisters' Image

In fact, the major shortcoming of fractal image encoding is the searching of domains for the individual ranges, which takes intensive time in the encoding stage. There are various efforts to reduce the search and thus enhancing the compression speed. We select one of those approaches to be used in the thesis, this approach - as given from (Ghosh et al., 2003) – improves the efficiency of image encoding when the fixed block partitioning method is used. It is a randomized approach, where the domain blocks are searched randomly for every range block. This approach is simple to implement and at the same time, it is proved that it gives much higher speedup compared with other methods. (Sobereano, 2000)

4.7 Image's Measures of Quality

Error measurement is an important aspect of determining the effectiveness of an image compression scheme. Obviously, we want to know how the decoded image is far from the original image, this can be obtained by one of the following techniques

I Correlation

The closeness between two digital images can be quantified in terms of correlation function. This measure computes the similarity between two images (Avcıbaş, 2001). One of the correlation commonly used functions is

$$Correlation = \frac{\sum_{x=1}^{H}\sum_{y=1}^{W} C'(x,y)\times C(x,y)}{\sum_{x=1}^{H}\sum_{y=1}^{W} C'(x,y)^2} \qquad \text{............ (4.8)}$$

II Peak Signal to Noise Ratio (PSNR)

The PSNR of an image is a typical measure used for evaluating image quality, the used PSNR function is described in equation 4.9. (Welsted, 1999)

$$PSNR = 20\log_{10}\frac{255}{RMSE} \qquad \text{............ (4.9)}$$

III Root Mean Square Error (RMSE): this technique was discussed in section 4.4.

CHAPTER FIVE

FRACTAL COLOR IMAGE COMPRESSION

This chapter discusses currently used algorithms for fractal color image compression and then, describes the components of the proposed compression system (the encoding and decoding systems).

5.1 Previous Work

Currently, a huge number of publications concentrate on the idea of fractal image compression for gray scale images and how to improve the efficiency and performance of its encoding process, while a very little attention was given for colored images; though in order to present a complete encoding scheme also color information has to be included.

The most straightforward method to encode a color image by gray level fractal image coding is to split the RGB color image into three channels, red, green, and blue, and compress the three channels separately by treating every color component's image as a gray scale image, as explained in figure 5.1. However, this method doesn't exploit the correlation between different color components resulting in relatively low compression ratio.

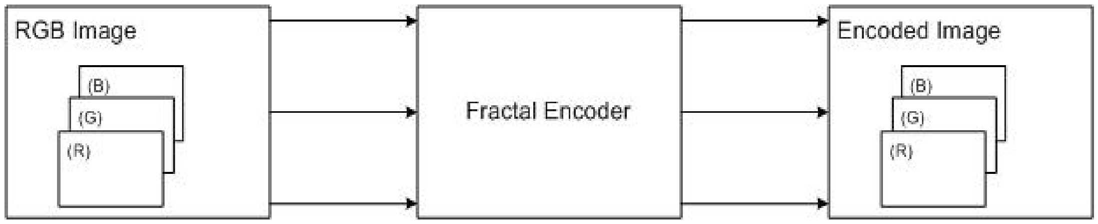


Figure 5.1: RGB Fractal Encoder

Figure 5.2 shows an example of using the three channels algorithm over a 128×128 image.

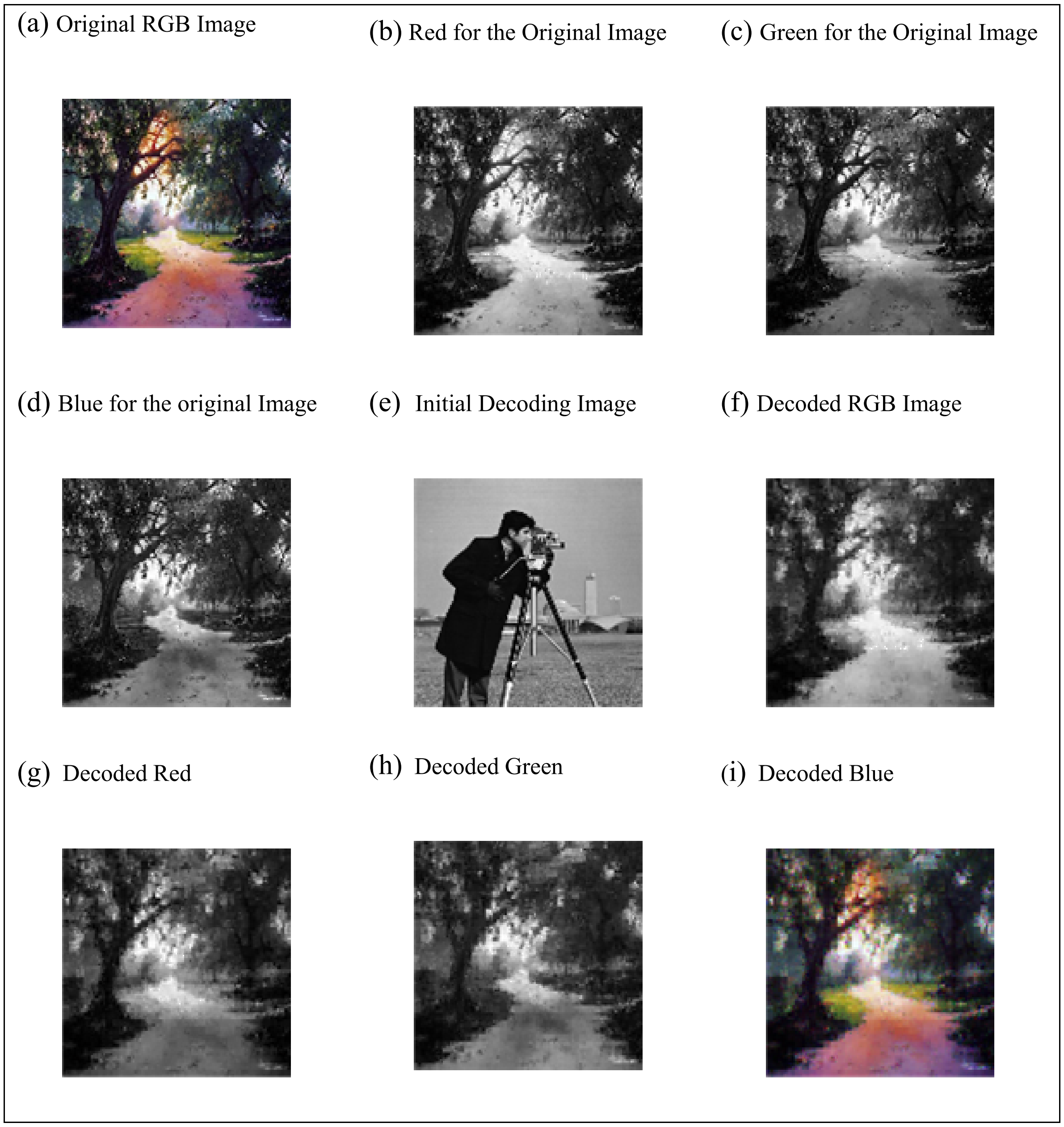

(a) Original RGB Image
(b) Red for the Original Image
(c) Green for the Original Image
(d) Blue for the original Image
(e) Initial Decoding Image
(f) Decoded RGB Image
(g) Decoded Red
(h) Decoded Green
(i) Decoded Blue

Figure 5.2: Applying three channels algorithm over a 128×128 image.

Other published algorithms in the area of fractal compression for colored images are: Master Plane algorithm (Brend et al., 1994), Vector Distortion algorithm (Zhang and Man, 1995), and Hierarchical IFS Coding (Nakano and Nakagawa, 2000).

The main idea of the Master Plane algorithm comes from the correlations between different color planes in the image. This algorithm is proved to be used for RGB and YUV images (because of the existence of highly correlated image planes especially for natural images). In the RGB color space the correlation coefficient between two of the planes is about 0.78 for blue-red, 0.89 for green-red, and 0.94 for green-blue plane.

As shown in figure 5.3, the encoding system for the Master Plane algorithm selects the green plane to be the master plane, while the red and blue planes represent the slaves. So, it applies the fractal encoder only for the green plane, while minimizing the contrast and brightness values for *R* and *B* planes without encoding them, and then it adds the calculated values of o_r, s_r, o_b, s_b to the green fractal code.

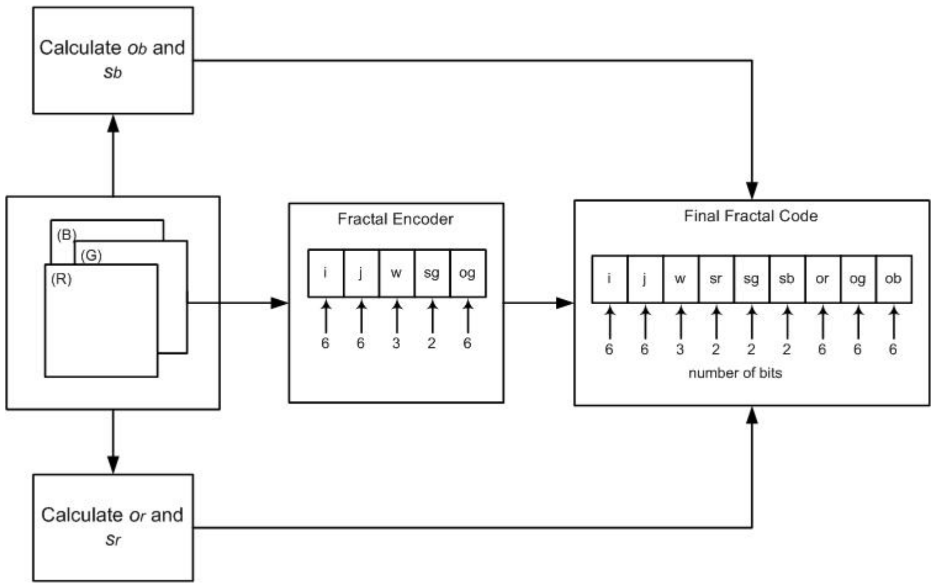


Figure 5.3: Encoding System of the Master Plane Algorithm.

In the decoding stage, the decoder constructs three fractal codes one for each plane. The three reconstructed fractal codes share the same values of *i*, *j*, and *w* that are taken from green plane fractal code, then appending o_r and s_r values to the red plane, the o_g and s_g values to the green plane, and finally the o_b and s_b values to the blue plane, as depicted in figure 5.4. (Zhang and Man, 1995)

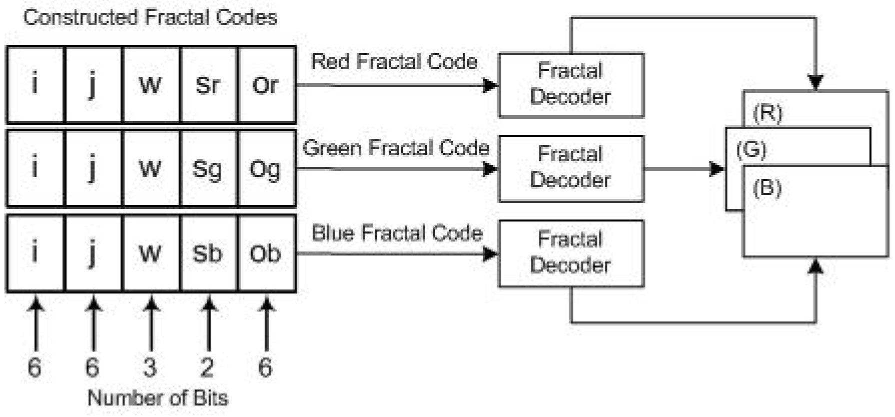


Figure 5.4: Master Plane Decoder

Figure 5.2 shows an example of using the Master Plane algorithm over a 128×128 image.

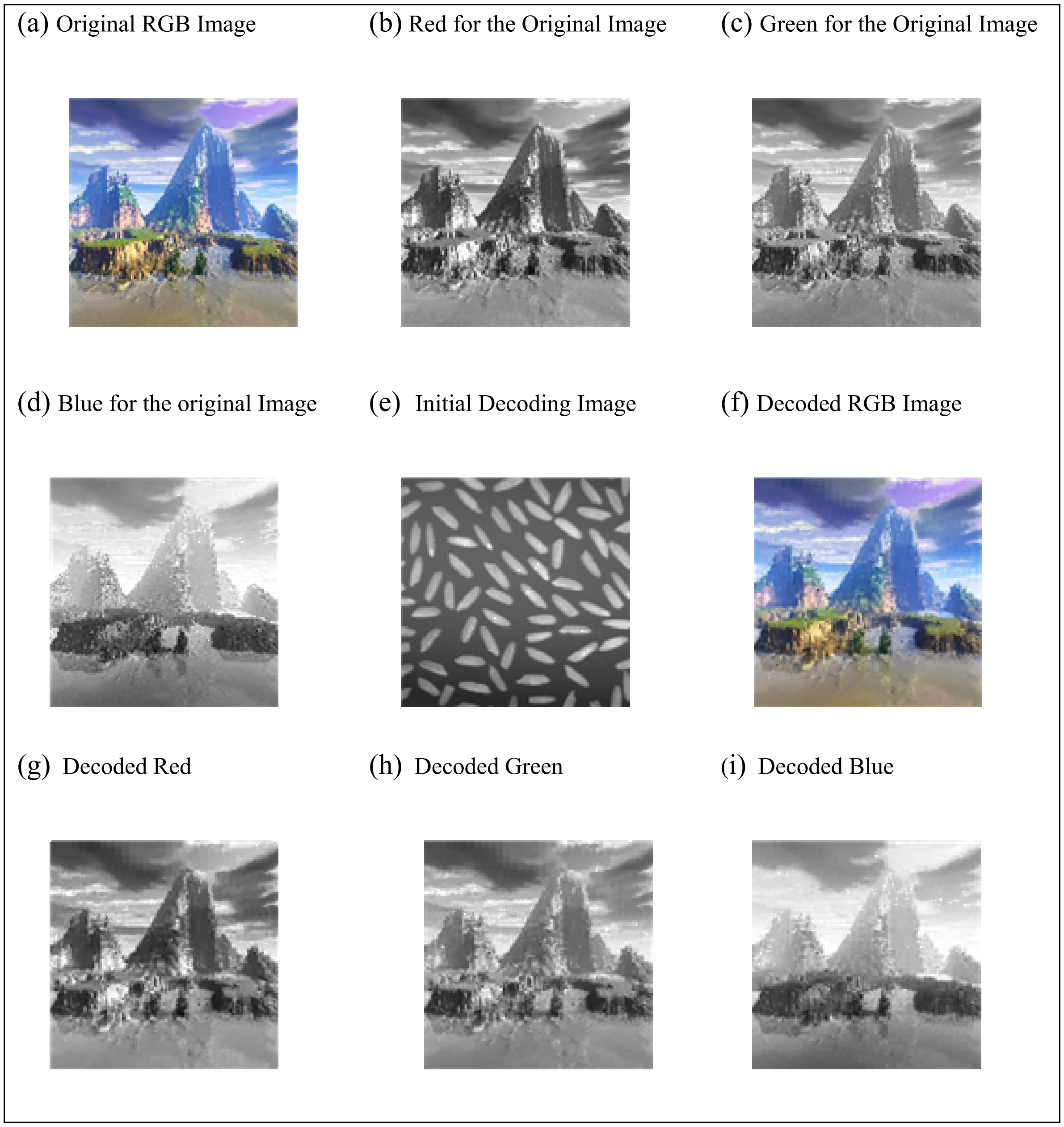

(a) Original RGB Image
(b) Red for the Original Image
(c) Green for the Original Image
(d) Blue for the original Image
(e) Initial Decoding Image
(f) Decoded RGB Image
(g) Decoded Red
(h) Decoded Green
(i) Decoded Blue

Figure 5.5: Applying Master Plane algorithm over a 128×128 image.

Color image compression using vector distortion measure is another method benefits from the correlations between color planes. This algorithm uses the three dimensional root mean square as a metric space; to exploit the spectral redundancy in RGB components. Note that, the fractal codes and the decoding system for this method have the same representation as that used in the Master Plane algorithm. (Zhang and Man, 1995)

The hierarchical IFS coding is one of the most recent methods used for coding color images. In fact this technique was directed toward efficiency improvement of fractal encoding for gray scale images, then they use this efficient method to encode the green plane to apply the master green algorithm (Nakano and Nakagawa, 2000).

5.2 Fractal Color Image Compression System

This section introduces the components of the proposed compression systems. This compression system is composed of: Fractal Image Encoding and decoding systems.

5.2.1 Fractal Image Encoding System

This section discusses briefly the primary units of the proposed encoding system. As depicted in figure 5.6, the encoding system is composed of three units: Pre-Processing unit, Fractal Image Encoder/s, and the Fractal File/s.

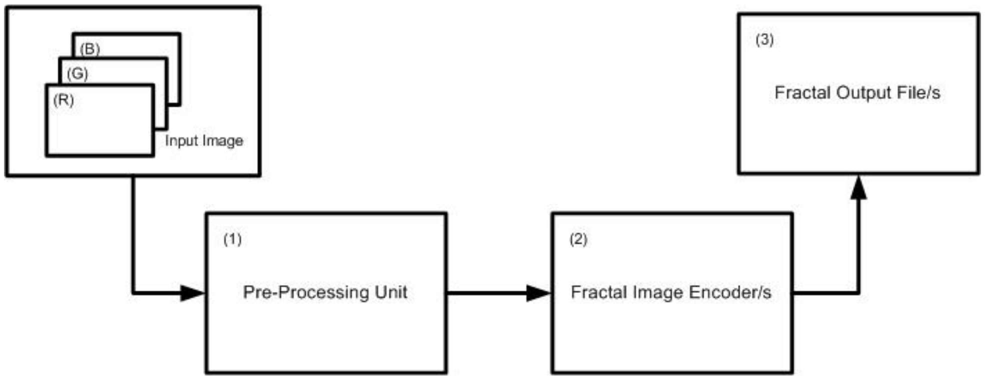


Figure 5.6: Fractal Encoding System

- *Pre- Processing Unit* is the simplest unit in the encoding system. It contains: image smoothing, conversion between color spaces, and it may contain color quantization for one or more planes. Low pass filters[*] are used for image smoothing in all proposed models. Conversion between RGB and HSV color spaces or image quantization techniques are used, as explained in the next chapter.

- *The Fractal Image encoder* is the same as that used for gray scale images.
- *Fractal Output File* is the file in which fractal codes are saved. In the Master Plane and Vector Distortion algorithms the final fractal codes are saved in one file, while in the three channels algorithm has three output files, one for each plane.

5.2.2 Fractal Image Decoding System

Now, let us discuss the primary units of the proposed decoding system. As depicted in figure 5.7, the decoding system is composed of two units: Fractal Image Decoder/s unit, Post-Processing unit.

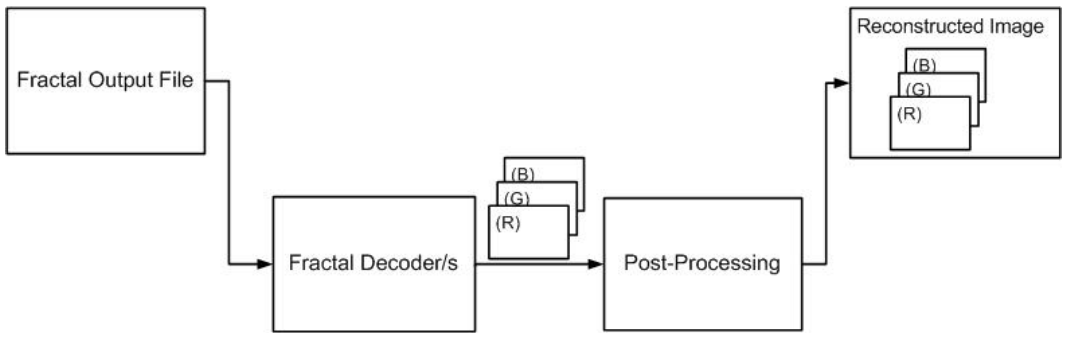


Figure 5.7: Fractal Decoding System.

- The *Fractal Image Decoder* is the same as that used for decoding gray scale images, as discussed in the previous chapter.

[*] The low pass filter that is used through out the thesis is the average filter.

- The *post-processing* unit, which may contain one or more of the following techniques:

1) *Dithering*: image dithering is a way to increase color resolution at the expense of losing spatial resolution. This means you can produce more colors from the available ones in your color map. The new colors are produced by placing pixels with different colors next together. In this case human eyes perceive a color which is the combination of the two colors. For example if you assign the red and blue beside each other, their color looks to be violet; this is because human eyes act like a filter and take the mean of the colors. In this case spatial resolution is lost; since two pixels have been used to produce one color. (Pins and Hild, 1989)
2) *Relaxation Filtering*: this type of filters -in its calculation- is like the average filter, where the gray level at each pixel is replaced with the average of the gray levels in a square neighborhood, *this is* called *moving average filter*. Average filters with very large squares called the relaxation filter. (Castleman, 1996)

CHAPTER SIX

FRACTAL IMAGE COMPRESSION FOR COLORED IMAGES USING BIOLOGICAL MODEL

In this chapter we will look at our proposed models, which are about compressing colored images using fractal encoding technique. Code samples about some of the proposed models are shown in appendix-C.

6.1 Reduced RGB Model

In 1985, Cawlishaw proposed that, since the human visual system is most sensitive to the green part of the spectrum, more resolution needs to be given to the green component than to the red or blue. He proposed a palette consisting of sixteen shades of green, four shades of blue, and four shades of red, as shown in figure 6.1. This palette provides a very simple transformation of 24-bit image to 8-bit image, where the four most significant bits represent the green component, the next two bits represent the red component, and the least two bits represent the blue component. This particular ordering of bits in each pixel has the advantage of representing a color image into one plane image (gray image).

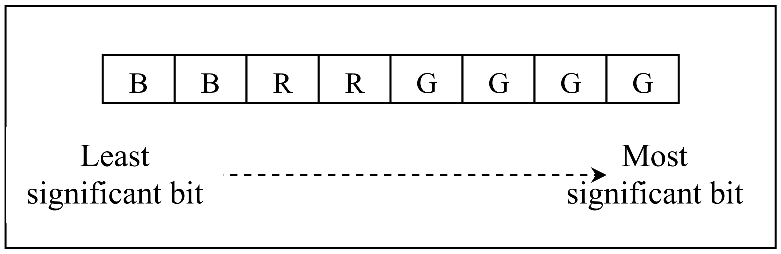


Figure 6.1: Bit Representation of a Pixel in the Reduced RGB Image

The Reduced RGB model is the first proposed model. It is called Reduced RGB; because the RGB image is reduced into one plane by using Cawlishaw's proposed palette. In this model after reducing the RGB model into one plane, we apply fractal image compression (block oriented fractal encoding, with block size = 4) to the reduced image, as shown in figure 6.2. This gives a compression ratio of 16:1.

In the decoding system, the inverse process has to be done, where the fractal decoded images will reconstruct the RGB image.

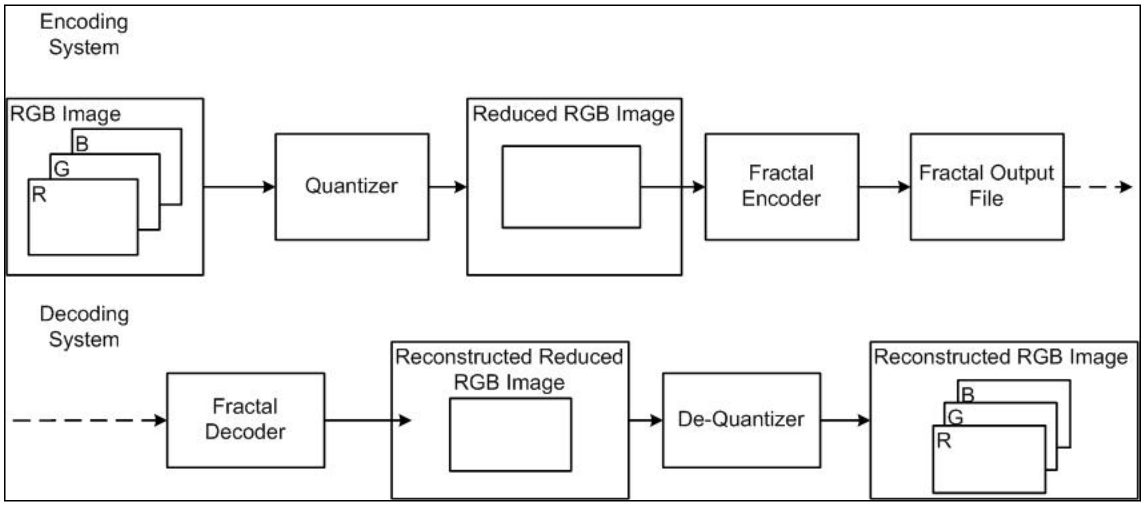


Figure 6.2: Reduced RGB Model

Unfortunately, this model doesn’t give good results when it is applied over natural and painted images. As shown in figures 6.3 and 6.4 respectively, the distortion becomes around edges.

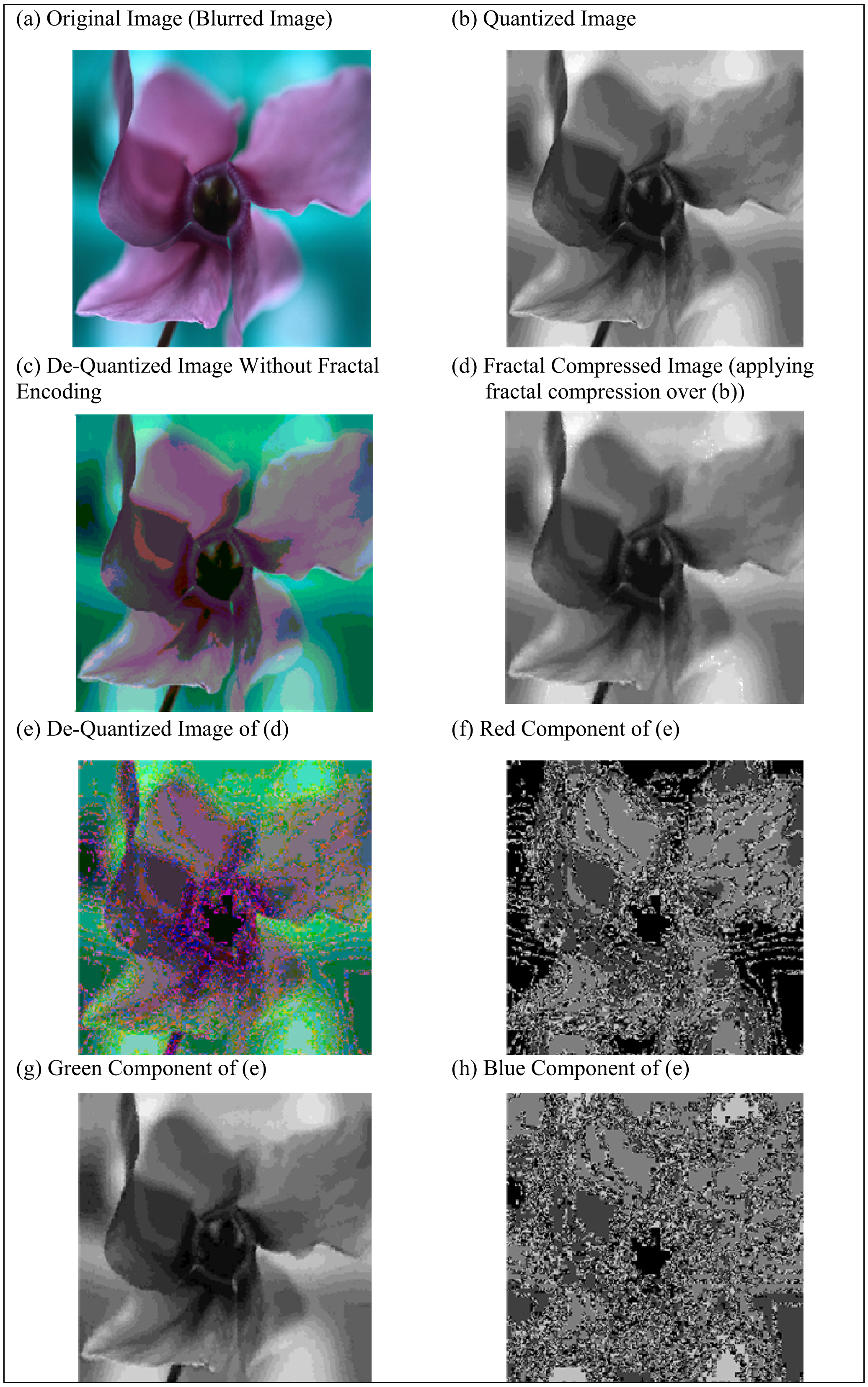

(a) Original Image (Blurred Image)
(b) Quantized Image
(c) De-Quantized Image Without Fractal Encoding
(d) Fractal Compressed Image (applying fractal compression over (b))
(e) De-Quantized Image of (d)
(f) Red Component of (e)
(g) Green Component of (e)
(h) Blue Component of (e)

Figure 6.3: Applying Reduced RGB Model over the purple flower image

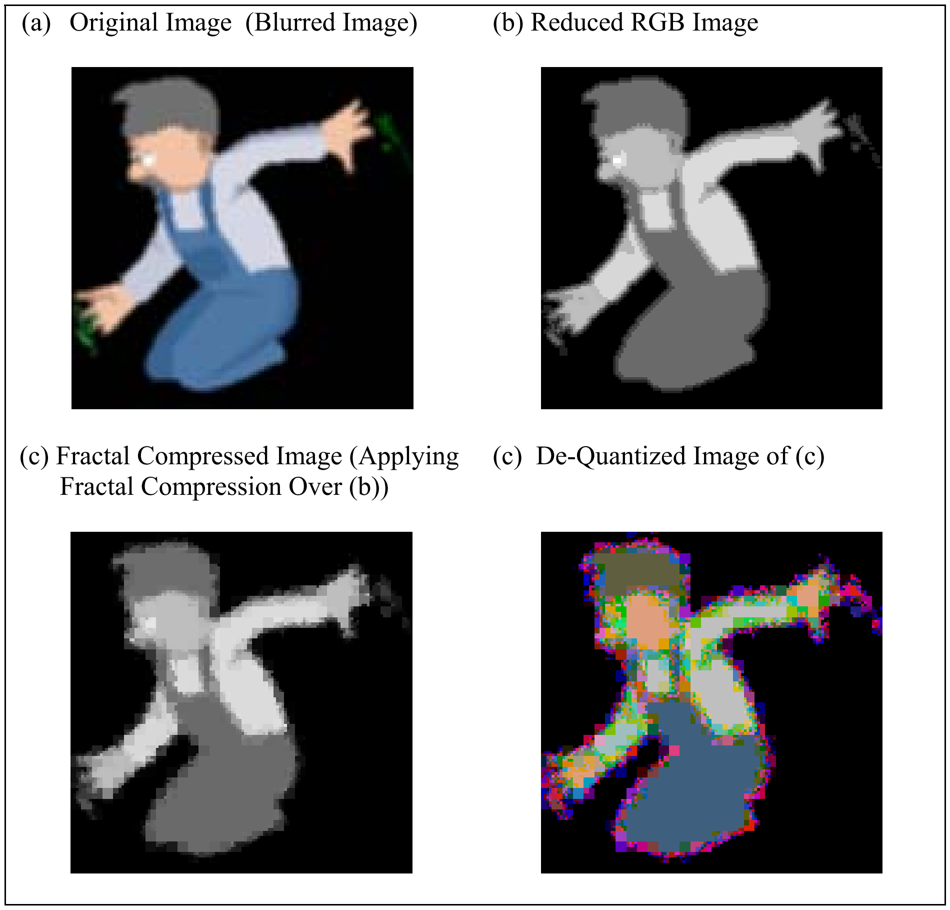

(a) Original Image (Blurred Image)
(b) Reduced RGB Image
(c) Fractal Compressed Image (Applying Fractal Compression Over (b))
(c) De-Quantized Image of (c)

Figure 6.4: Applying Reduced RGB Model over Farmer Image (Painted Image)

6.2 Reduced HSV

The Reduced HSV model is the same as the Reduced RGB model, but, there is a slight difference between them, it is the use of the HSV color space instead of the RGB color space, as depicted in figure 6.5.

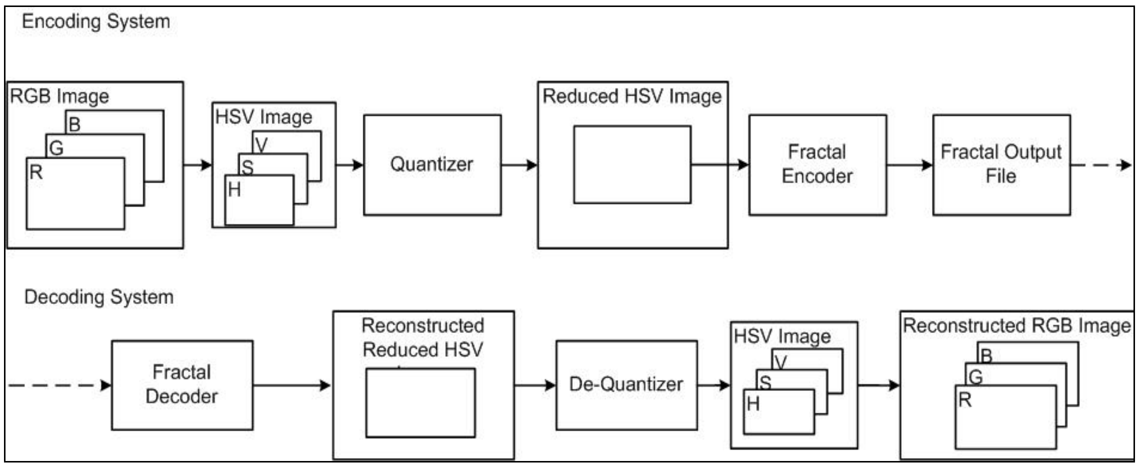


Figure 6.5: Reduced HSV Model

Pay attention that the generation of the reduced HSV image is different than Cawlishaw's proposed palette, since it uses three bits to represent the V plane, two bits to code the S plane, and the last three bits are used to code the H plane, their order as depicted in figure 6.6.

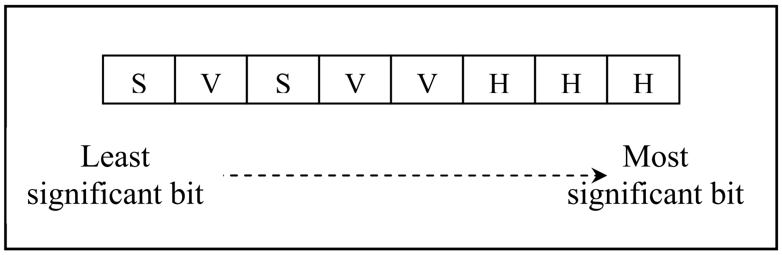


Figure 6.6: Bit Representation of a Pixel in the Reduced HSV Image

The results of this model are worse than the results of the Reduced RGB model, because it is more sensitive near edges, this is shown in the figures 6.7 and 6.8 respectively. On the other hand, images' colors in this model are less sensitive to the quantization process compared with the Reduced RGB model; this is due to the use of the HSV model which separates color information in the H plane.

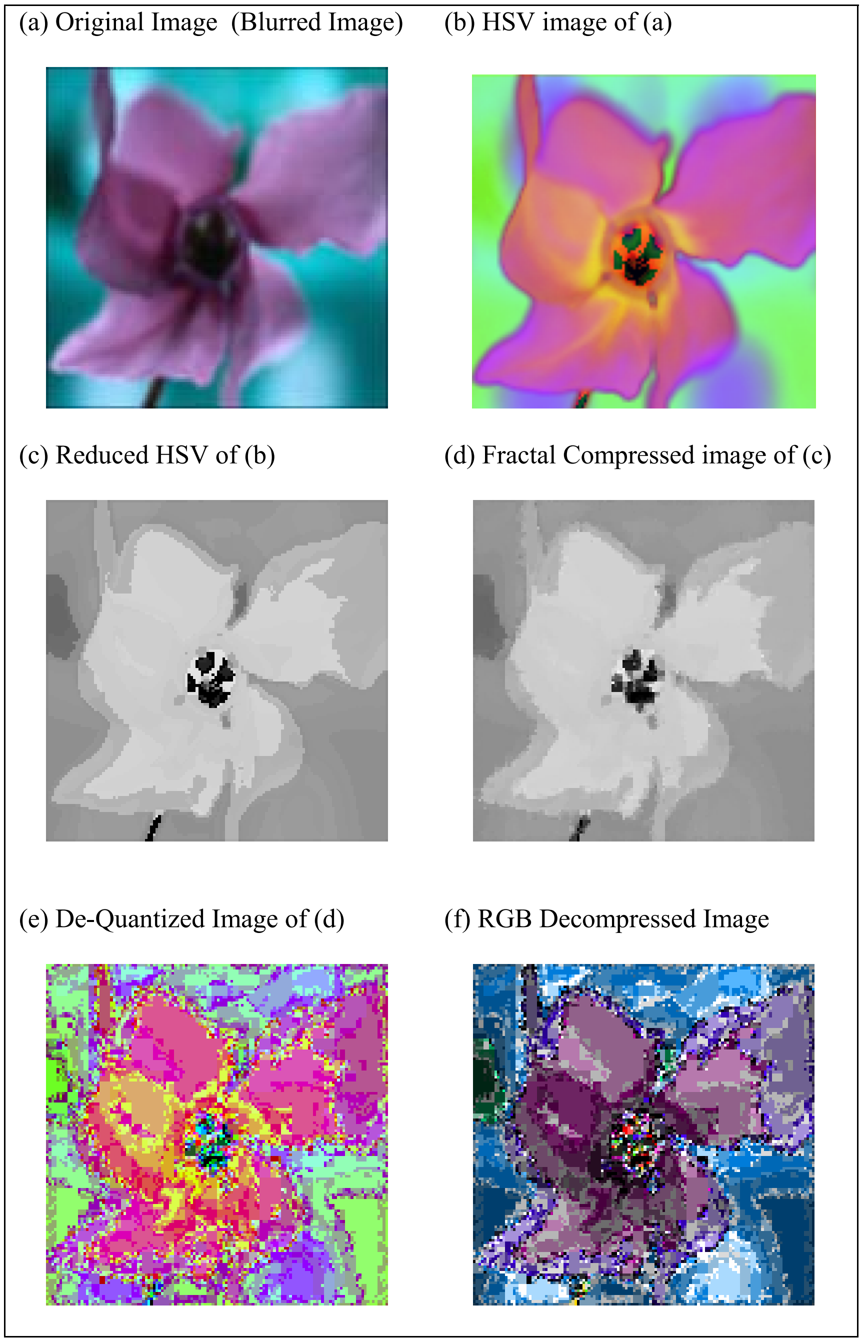

(a) Original Image (Blurred Image)
(b) HSV image of (a)
(c) Reduced HSV of (b)
(d) Fractal Compressed image of (c)
(e) De-Quantized Image of (d)
(f) RGB Decompressed Image

Figure 6.7: Applying the Reduced HSV Model to Purple Flower Image

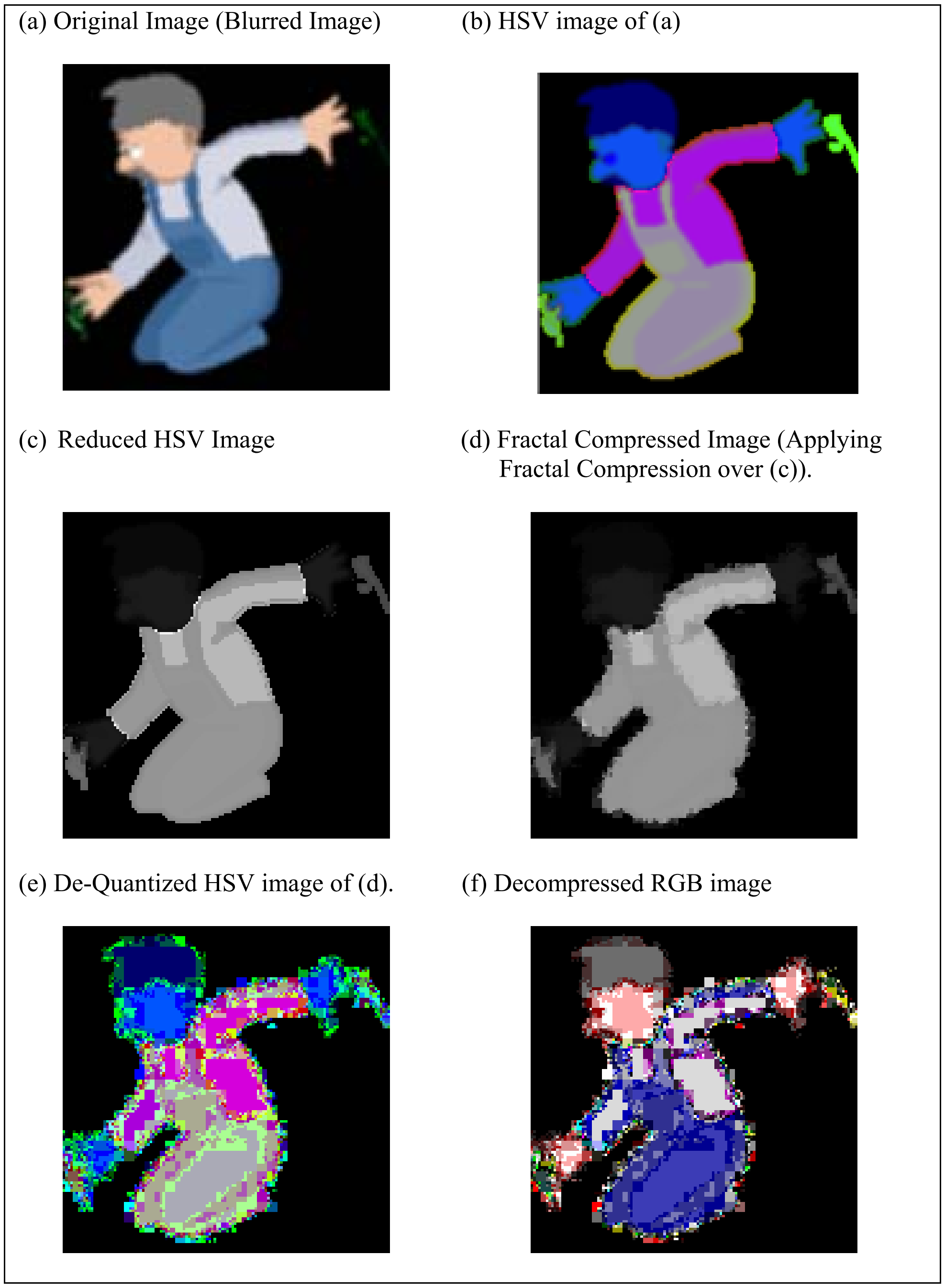

(a) Original Image (Blurred Image)
(b) HSV image of (a)
(c) Reduced HSV Image
(d) Fractal Compressed Image (Applying Fractal Compression over (c)).
(e) De-Quantized HSV image of (d).
(f) Decompressed RGB image

Figure 6.8: Applying Reduced HSV Model to the Farmer Image.

6.3 Reduced SV Model

Here, we try to reduce the S and V planes of the HSV image into one plane, the bit representation of the reduced image is as shown in figure 6.9.

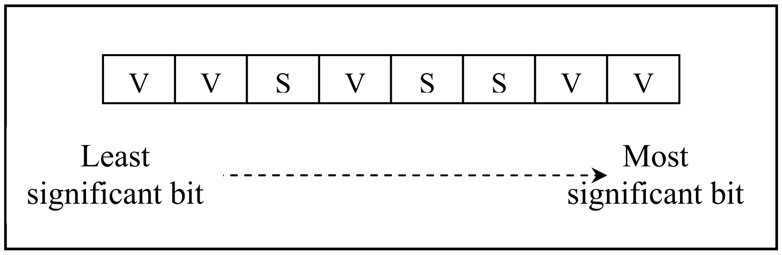


Figure 6.9: Bit Representation of a Pixel in the Reduced SV Image

As shown in figure 6.10, we have to apply the fractal encoder (block oriented fractal encoder, with block size = 4) twice, one for the H plane and the other for the reduced SV plane, this gives a compression ratio of 8:1.

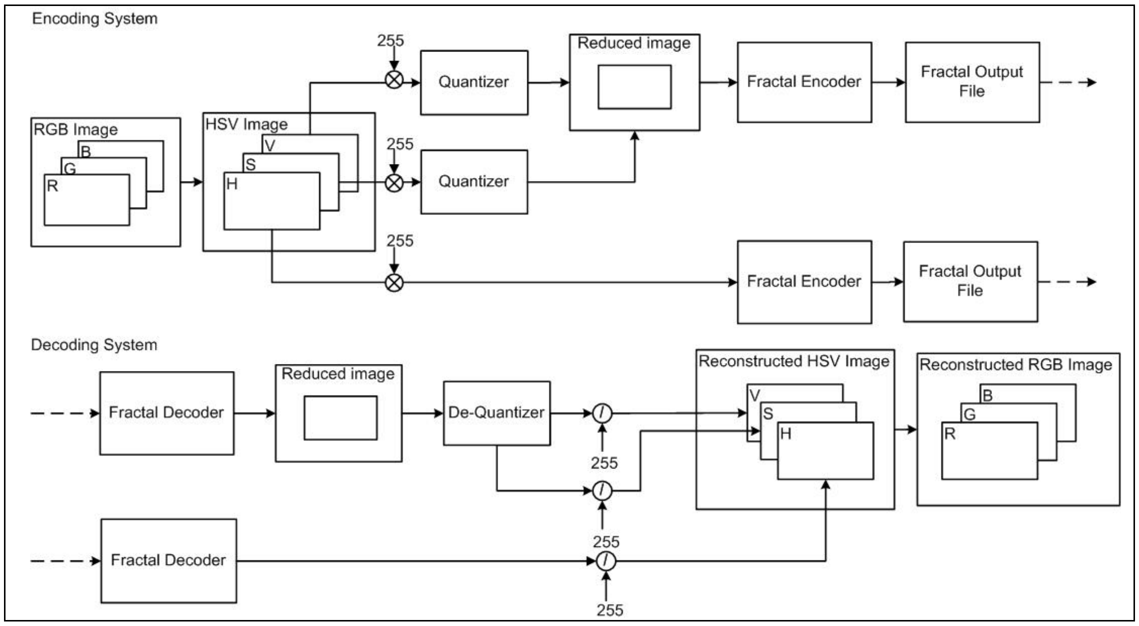
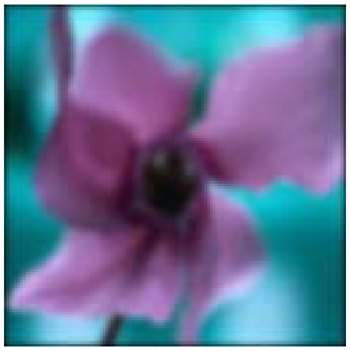


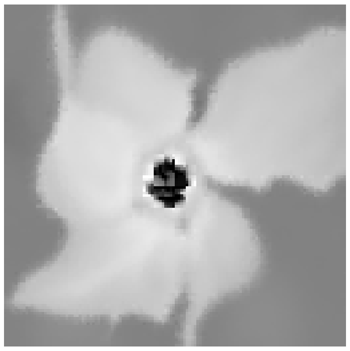
Figure 6.10: Reduced SV Model

In spite of having better results, a lower compression ratio is achieved, as shown in figure 6.11. We will achieve a better compression ratio with a less quality image, if we increase the size of the fractal encoder's range block (either for H plane or the reduced SV Plane). I.e.: the better the compression ratio the worse the quality of the image.

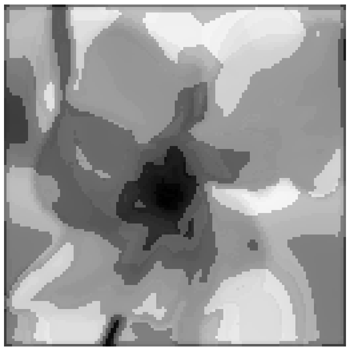
(a) Original Image (Blurred Image)



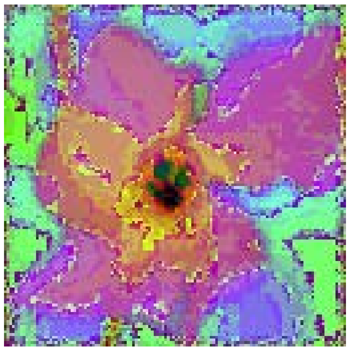
(b) Fractal Compressed H Plane



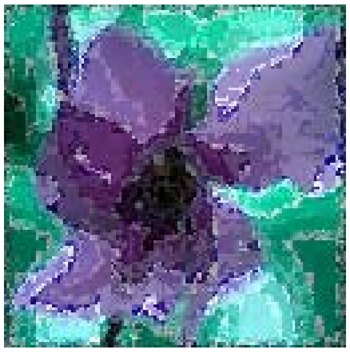
(c) Fractal Compressed Reduced-SV Plane



(d) HSV Decompressed Image (it is Composed of the De-Quantized SV Planes and H Plane)



(e) RGB Decompressed Image



(f) Decompressed Image after Applying Dithering over (e)

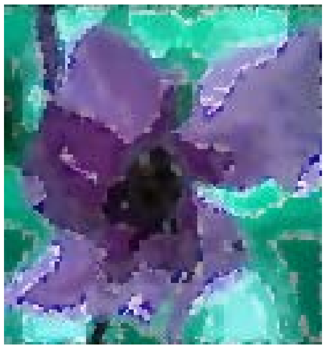


Figure 6.11: Applying SV Model to the Purple Flower Image

6.4 Reduced HV Model

The Reduced HV model reduces the H and V planes into one plane, where its representation code is as given in figure 6.12.

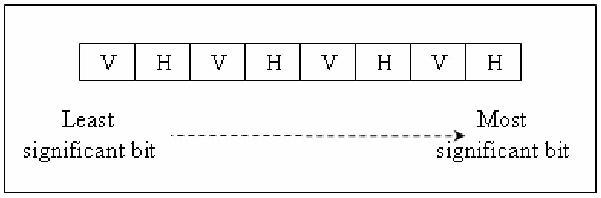


Figure 6.12: Bit Representation of a Pixel in the Reduced HV Image

As shown in figure 6.13, the fractal encoder is applied twice, one for S plane and the other one for the reduced HV plane. This model benefits from the property of the S plane that it has low information content. So, the fractal compression to this plane was done by partitioning the image into bigger size range blocks (8 or 16 for a 128×128 image), this gives a higher compression ratio. On the other hand, the reduction of the HV plane produces a big distortion in both the H and V planes, as illustrated in figure 6.14.

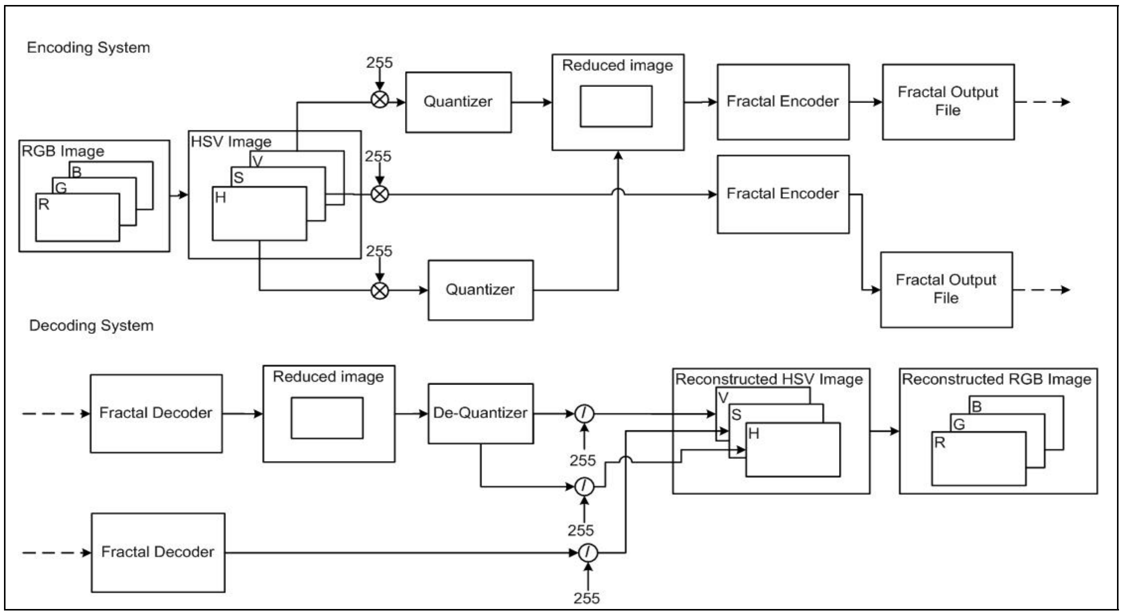
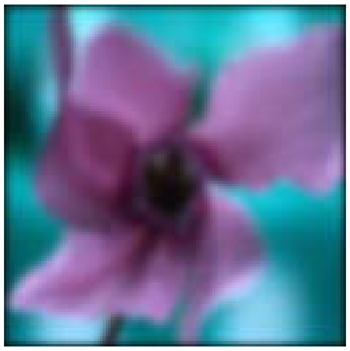
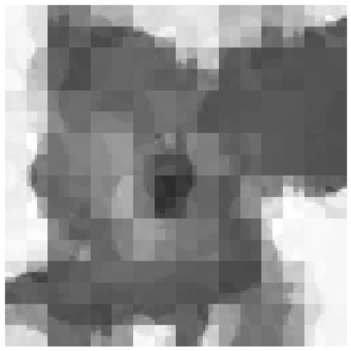


Figure 6.13: The Reduced HV Model

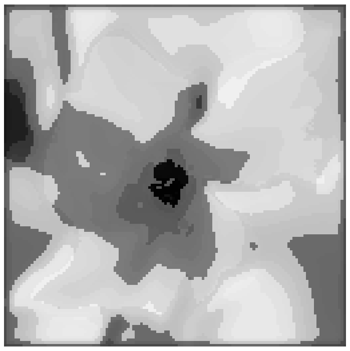
(a) Original Image (Blurred Image)



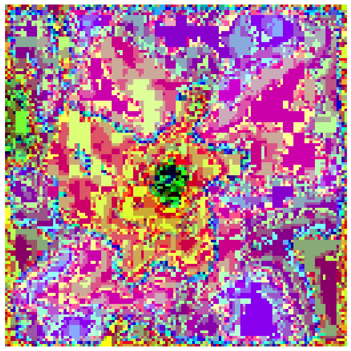
(b) Fractal Compressed S Plane



(c) Fractal Compressed Reduced-HV Plane



(d) HSV Decompressed Image (it is Composed of the De-Quantization of (c) and (b))



(e) RGB Decompressed Image

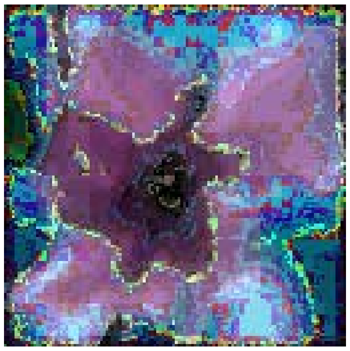


Figure 6.14: Applying the Reduced HV Model to the Purple Flower Image

6.5 Reduced HS Model

Another issue called the Reduced HS Model was proposed to reduce the H and S planes, its structure and bit representation are as shown in figures 6.15 and 6.16 respectively.

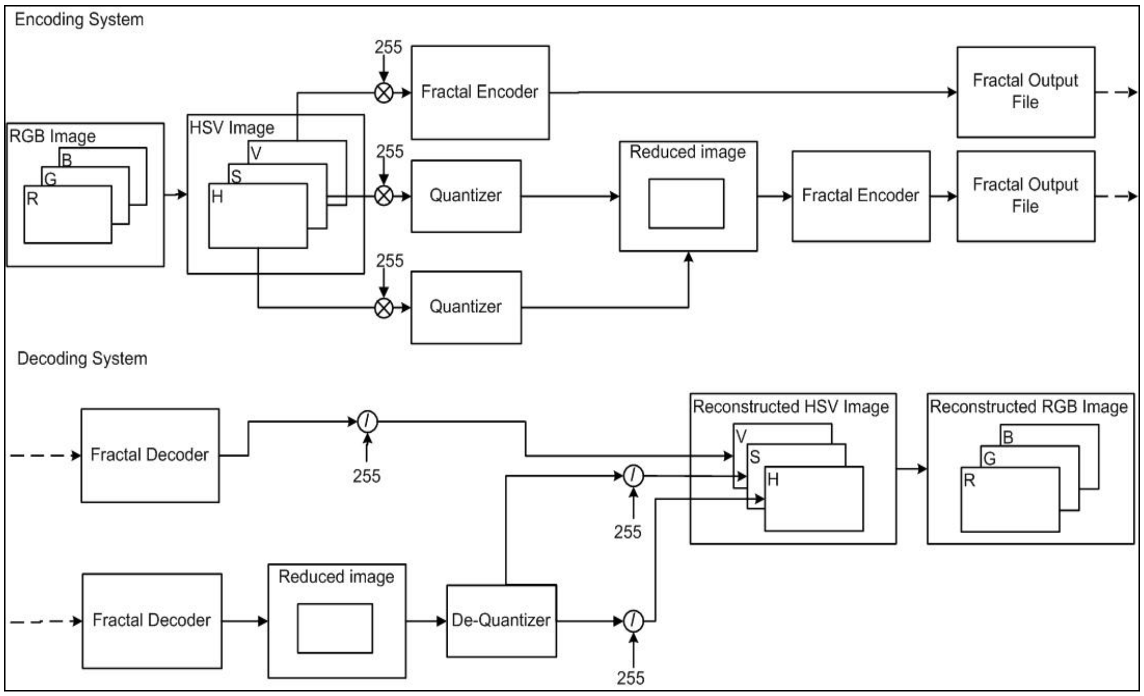


Figure 6.15: Reduced HS Model

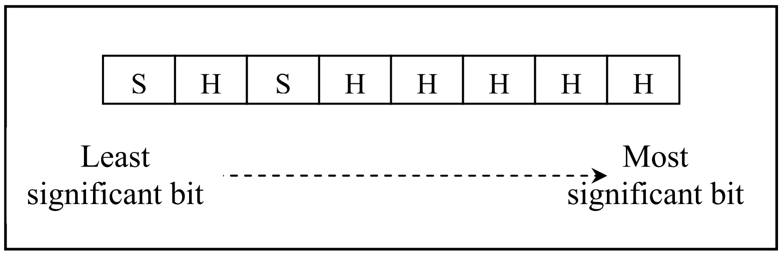


Figure 6.16: Bit Representation of a Pixel in the Reduced HS Image

As depicted in figure 6.17, we have achieved better results, but its compression ratio is still low (8:1), that's because the size of range blocks used in the fractal encoder which equals 4. So, if we think about increasing range block's size to encode the V plane; to increase the compression ratio, we must be careful; because it will affect the details of the image, while increasing the size of the range block to fractal encode the Reduced-HS image produces bad results, that is the image affected by the quantization and fractal encoding errors.

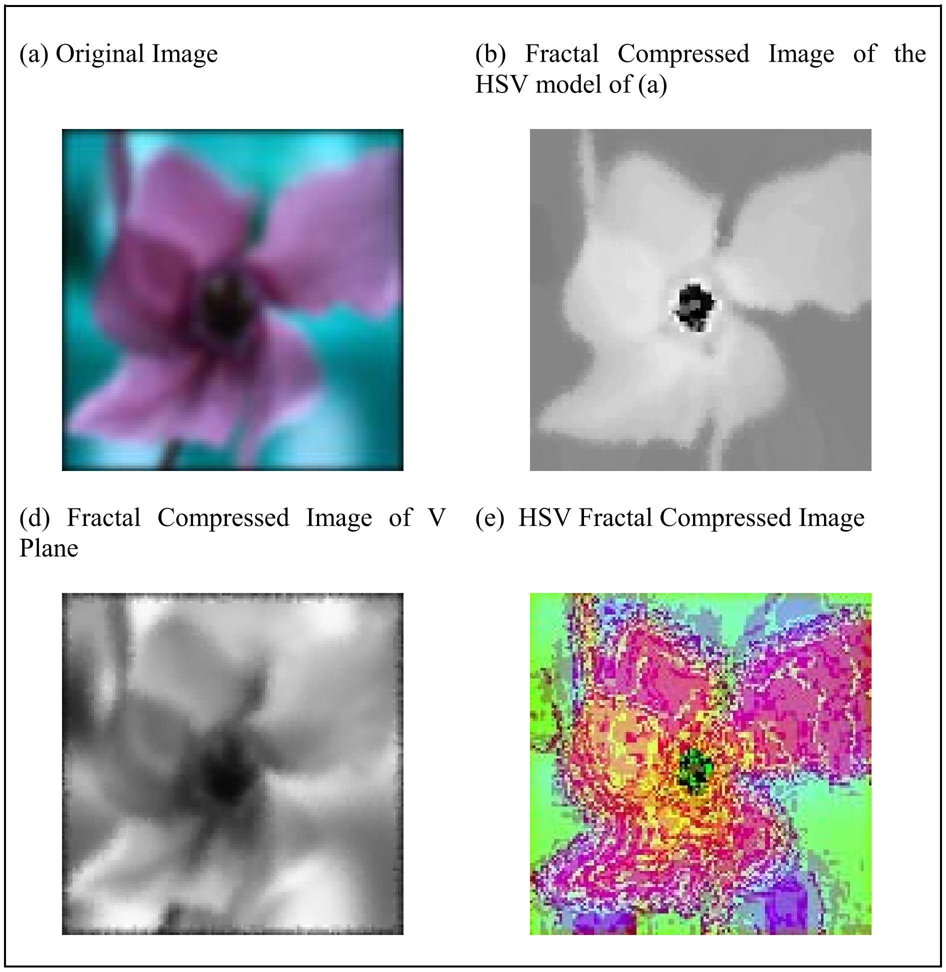

(a) Original Image
(b) Fractal Compressed Image of the HSV model of (a)
(d) Fractal Compressed Image of V Plane
(e) HSV Fractal Compressed Image

Figure 6.17: Applying HS Model to the Purple Flower Image

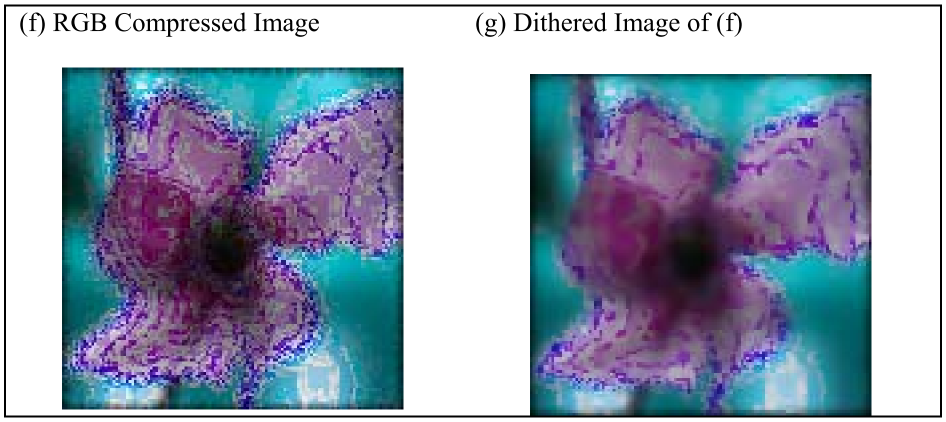


Figure 6.17: (Continued).

6.6 Three-Channel-HSV-Model (with Different Fractal Encoders)

After many experiments to combine the best efficiency, quality and compression ratio we create the Three-Channel-HSV-Model which is considered to be the best one among all previous models.

As shown in figure 6.18 instead of using the quantization (to reduce some planes in one plane) the fractal encoding is applied over the three planes separately. Note that, the fractal encoders don't share the same size of the range blocks used to encode each plane. The S plane has a low information content; so it could be encoded using larger size of range blocks, and since many of the natural images don’t have sharp variations in colors we can also use large blocks to encode the H plane. In images that don’t have a lot of details (edges), we can increase the size of the range block to encode the V plane.

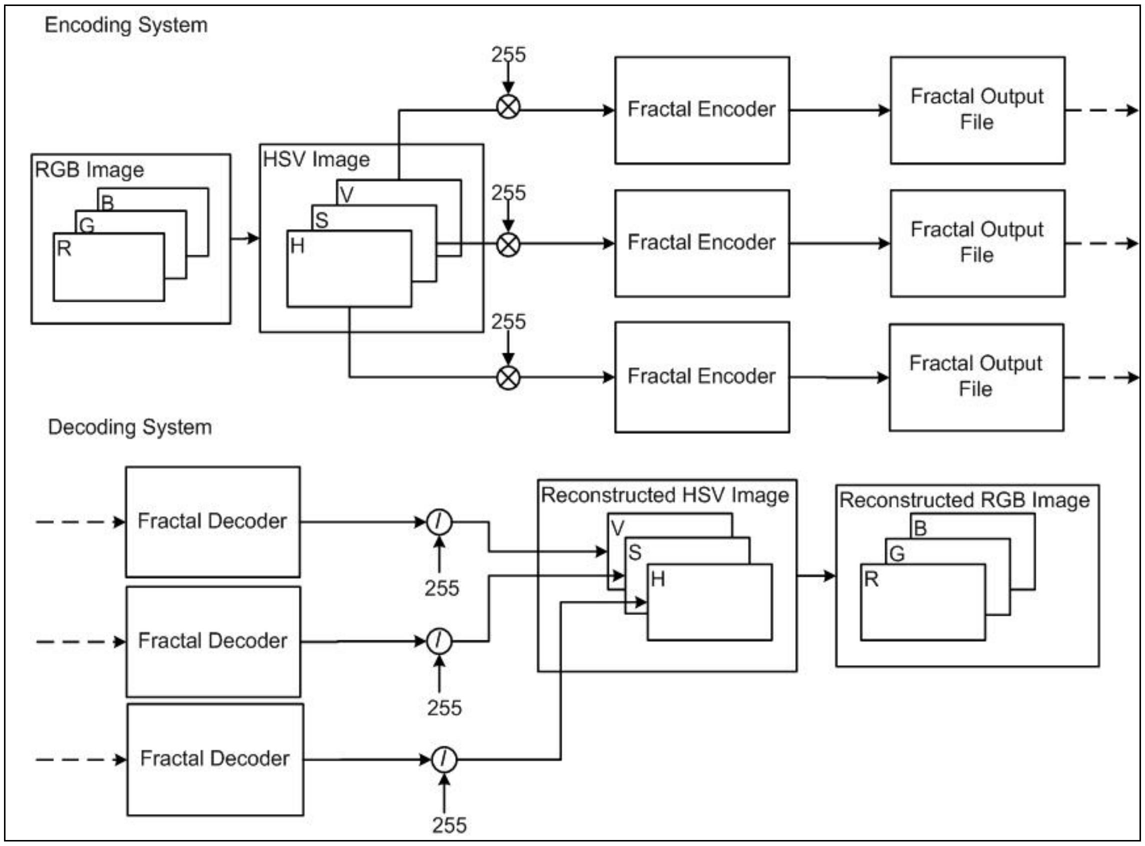


Figure 6.18: Three-Channel-HSV-Model with Different Fractal Encoders

Briefly, the separation between the image itself and its color information is the first advantage of the HSV model, where the low information content in the S plane guides to high compression ratio. This model works well especially in colored images which have low variation in colors; in this case a higher compression ratio will be achieved in the H plane, without affecting other models.

Up to now, we have covered all the aspects of our proposed models except the results and analysis of the *Three-Channel-HSV-Model*, which are the main subject of the next chapter.

CHAPTER SEVEN

THE THREE-CHANNEL-HSV-MODEL

RESULTS AND ANALYSIS

7.1 Experimental Results

In this chapter two examples of applying The Three-Channel-HSV-Model were discussed, where in the first example, a harmonic image was chosen and in the second one an image with high details and large number of colors was selected, as shown in figure 7.1.

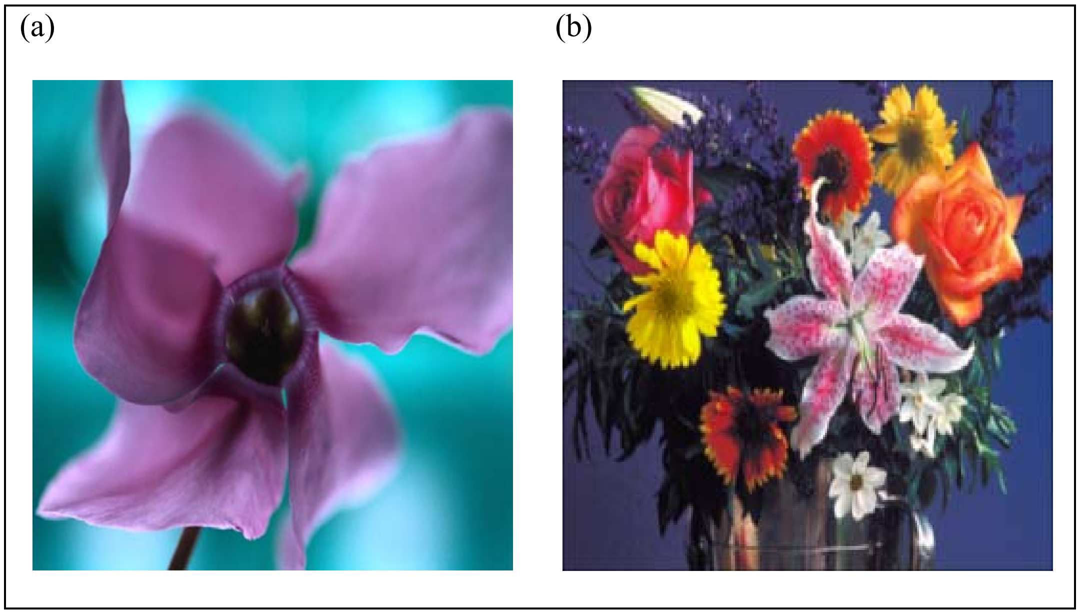


Figure 7.1: Testing Images

As shown in figures 7.2-(1) and 7.2-(2), we have image "a" and its corresponding HSV image. Applying fractal encoding for H, S and V planes separately, using range blocks of size 4 to encode H and V planes, while varying the size of the range blocks for S plane. As you see in figures 7.2-(3), 7.2-(5), the size of the S range blocks doesn’t make deference.

As shown in figures 7.2-(7), 7.2-(9), and 7.2-(11), increasing the size of the range blocks for H and V will reduce the encoding time and result a high compression ratio, while some distortions appear around edges, which could be treated using dithering technique.

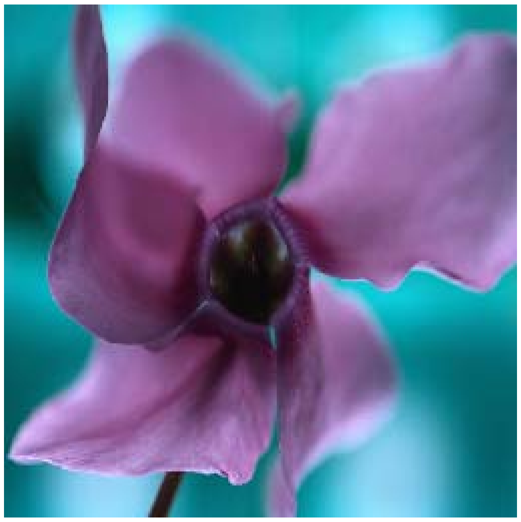
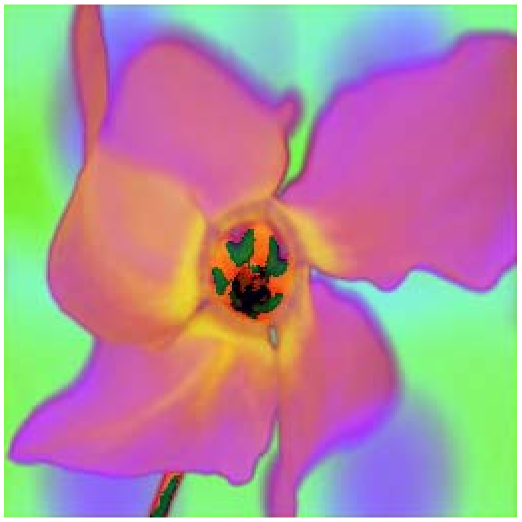
Image "a"	
1) Original RGB Image 	2) Original HSV Image 

Figure 7.2: Applying Three-Channel-HSV-Model for Image "a"

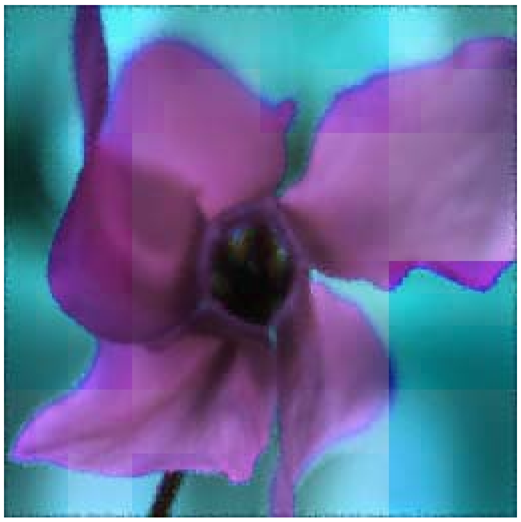
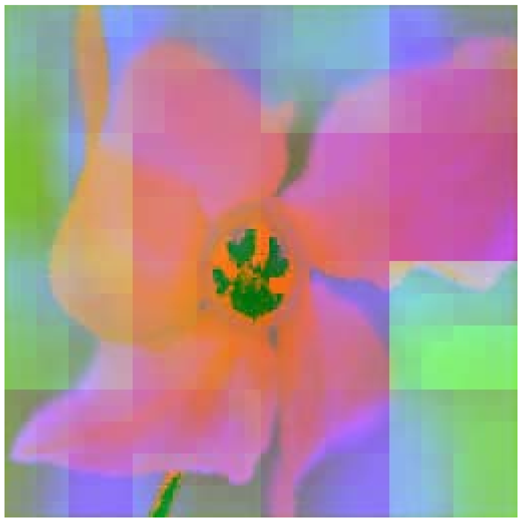
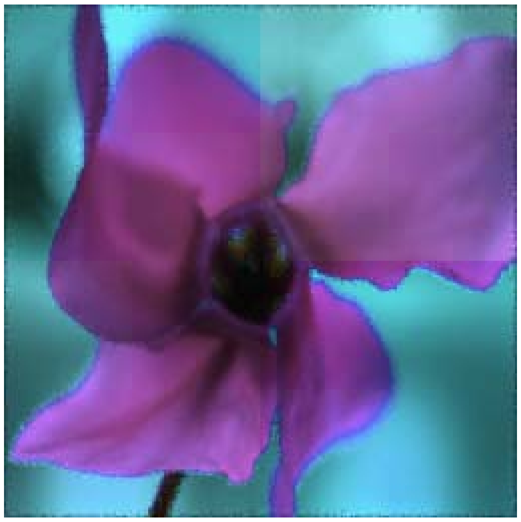
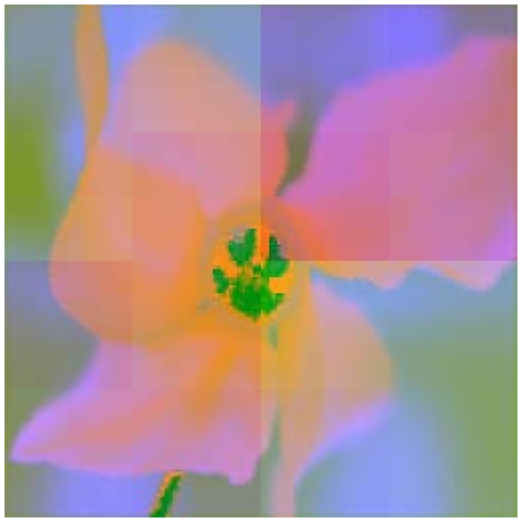
Size of the range blocks: H = 4, S = 64, and V = 4	
3) RGB Decompressed Image 	4) HSV Decompressed Image 
Size of the range blocks: H = 4, S = 128, and V = 4	
5) RGB Decompressed Image 	6) HSV Decompressed Image 

Figure 7.2: (Continued)

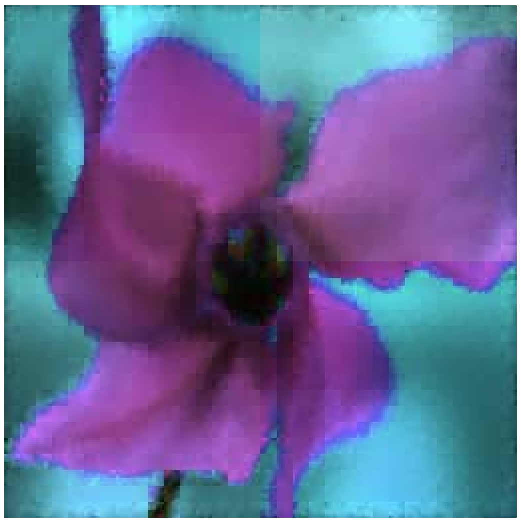
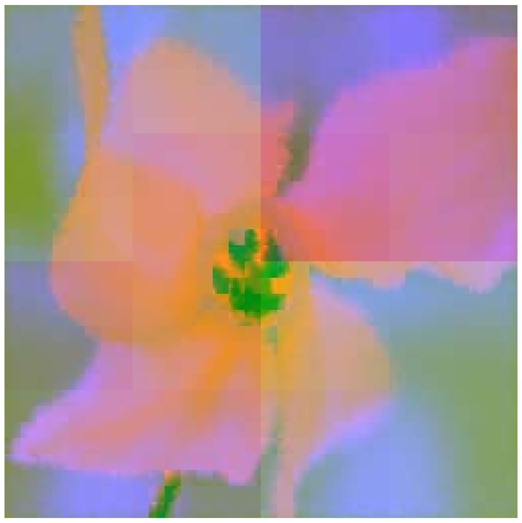
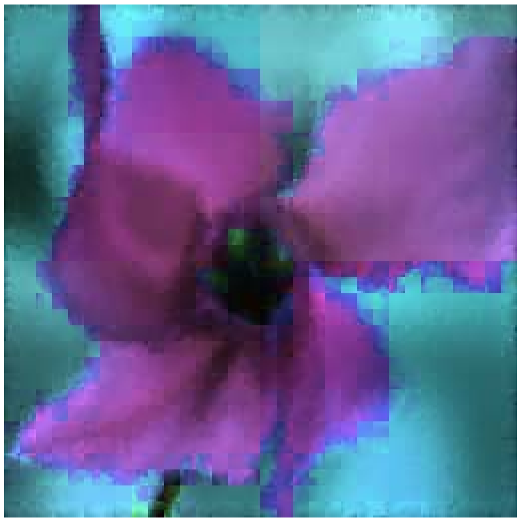
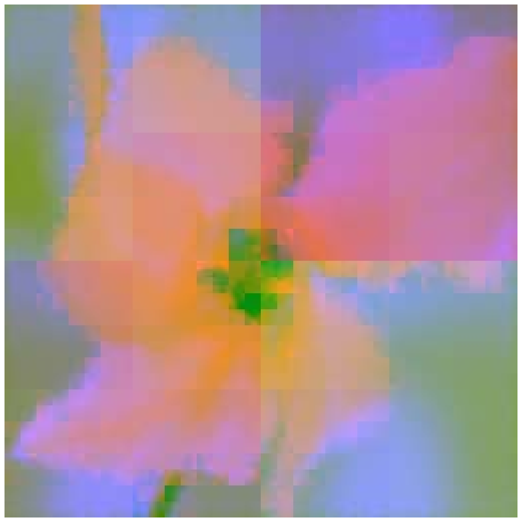
Size of the range blocks: H = 8, S = 128, and V = 8	
7) Decompressed RGB Image 	8) Decompressed HSV Image 
Size of the range blocks: H = 16, S = 128, and V = 8	
9) Decompressed RGB Image 	10) Decompressed HSV Image 

Figure 7.2: (Continued)

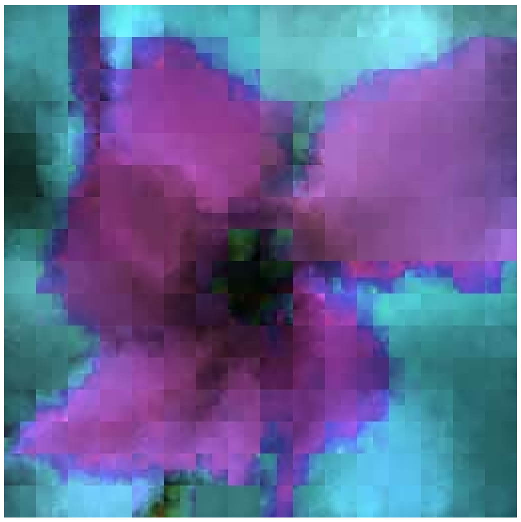
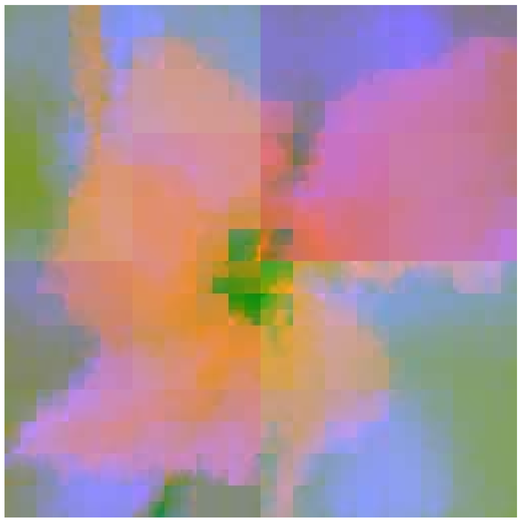
Size of the range blocks: H = 16, S = 128, and V = 16	
11) Decompressed RGB Image 	12) Decompressed HSV Image 

Figure 7.2: (Continued)

Figure 7.3 summarizes the values of RMSE and PSNR for the previous results, as you see, the quality of the image in the case of "1" and "2" is very close according to the PSNR values; this means that the saturation doesn't make a large difference.

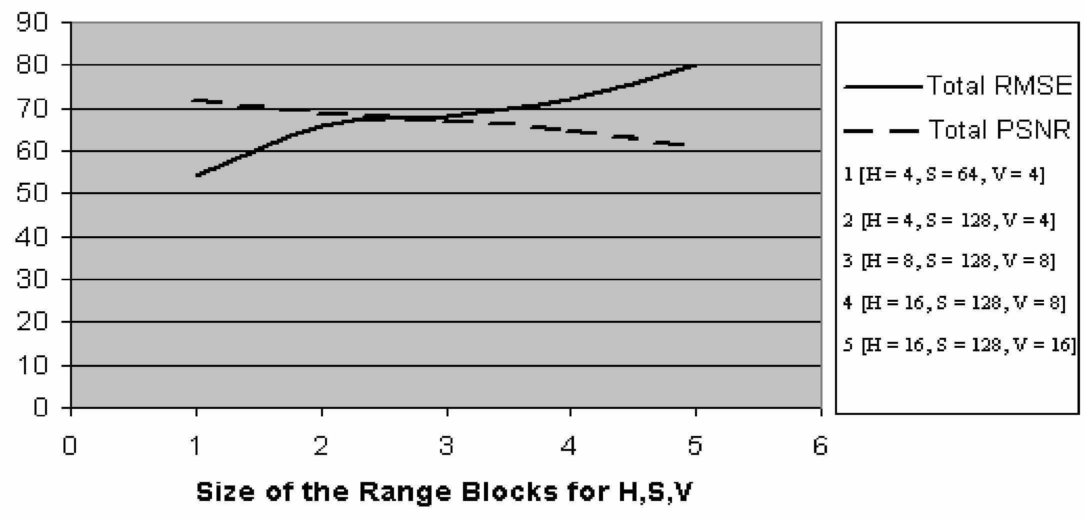


Figure 7.3: RMSE and PSNR values for Image "a"

Table 7.1 shows the quality of the decompressed image "a" compared with the original image "a" under several situations, where it has variations in the size of the H, S and V range blocks. As you see, the blue plane has the highest correlation values; clearly, this is due to the color of image "a", it is obvious that it has a bluish color, so if you think about it in the RGB Model, to get a high quality image you have to encode the blue plane in the highest quality. Also, the values of the PSNR and RMSE for the blue plane don't change in the first two columns, where we change the size of the S range blocks.

Table 7.1: RMSE, PSNR and Correlation between image "a" and the reconstructed images using the Three-Channel-HSV-Model for different values of H, S and V.

Image "a"					
[H S V]	[4 64 4]	[4 128 4]	[8 128 8]	[16 128 8]	[16 128 16]
Red Correlation	0.91	0.86	0.85	0.83	0.82
Green Correlation	0.94	0.93	0.93	0.92	0.90
Blue Correlation	0.99	0.99	0.98	0.98	0.96
Red RMSE	26	34	34	36	37
Green RMSE	20	24	25	26	28
Blue RMSE	8	8	9	10	15
Total RMSE	54	66	68	72	80
Red PSNR	20	18	18	17	17
Green PSNR	22	21	20	20	19
Blue PSNR	30	30	29	28	25
Total PSNR	72	69	67	65	61

Applying the same model for image "b" – which has high details and large variations in colors - doesn't imply good results, as shown in figure 7.4. More results are shown in appendix-A and appendix-B respectively.

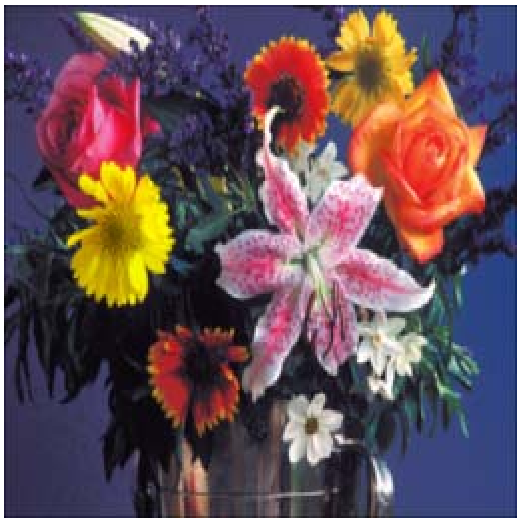
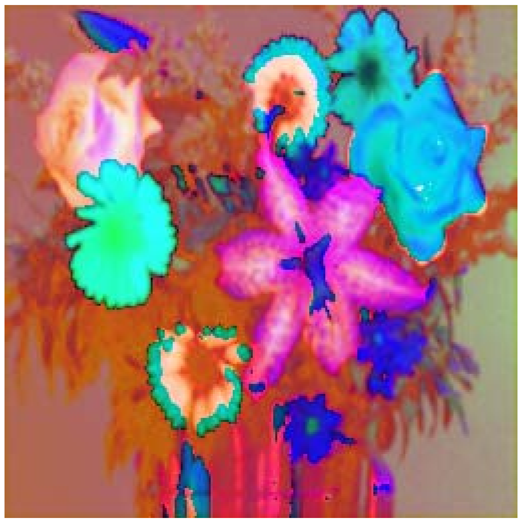
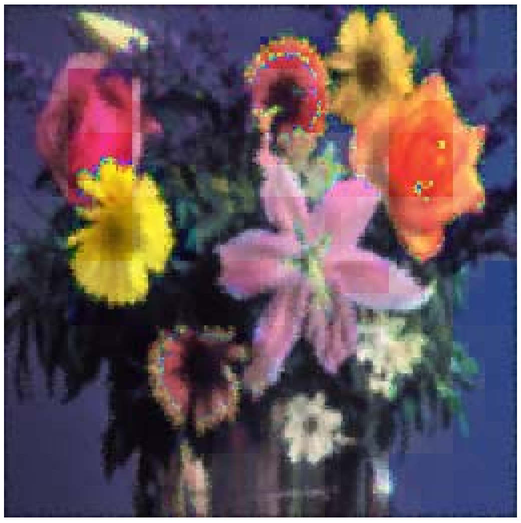
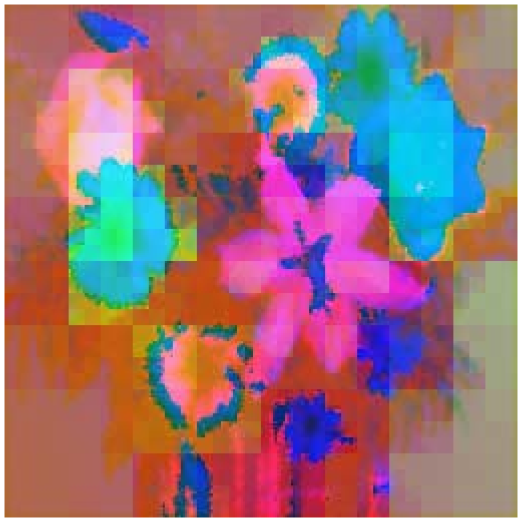
Image "b"	
1) Original RGB Image 	2) Original HSV Image 
Size of the range blocks: H = 4, S = 32, and V = 4	
3) RGB Decompressed Image 	4) HSV Decompressed Image 

Figure 7.4: Applying Three-Channel-HSV-Model for Image "b"

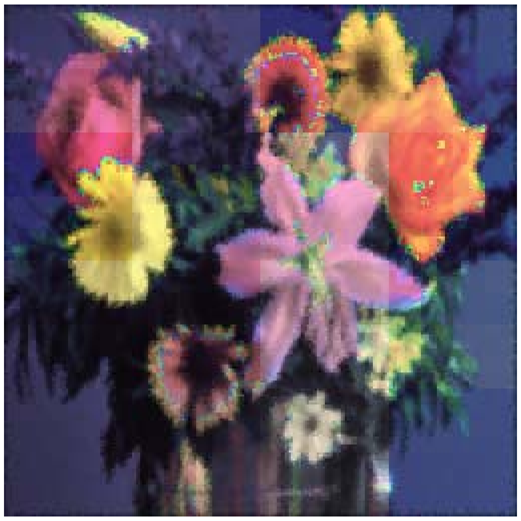
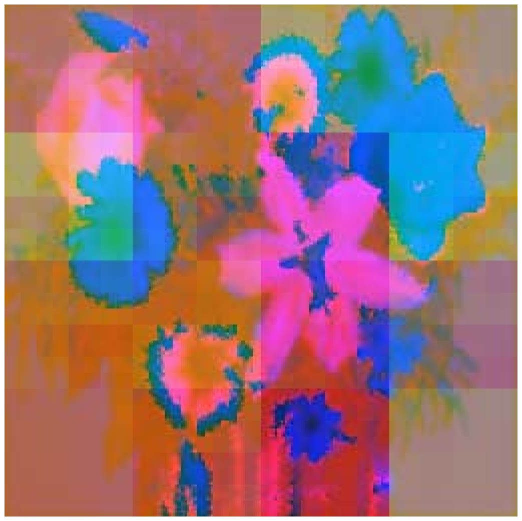
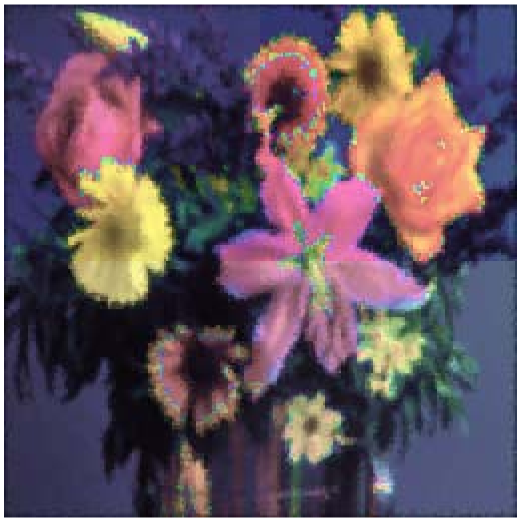
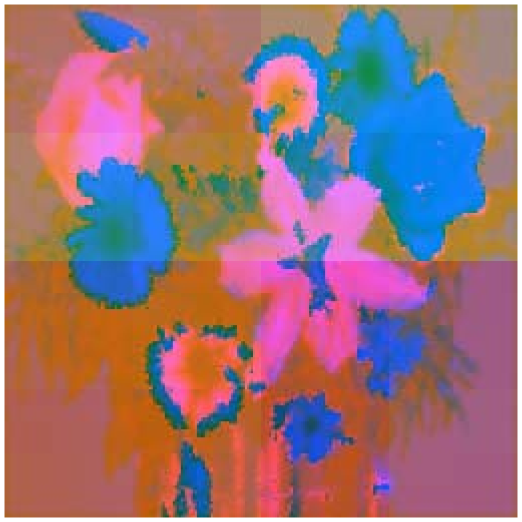
Size of the range blocks: H = 4, S = 64, and V = 4	
5) RGB Decompressed Image 	6) HSV Decompressed Image 
Size of the range blocks: H = 4, S = 128, and V = 4	
7) RGB Decompressed Image 	8) HSV Decompressed Image 

Figure 7.4: (Continued)

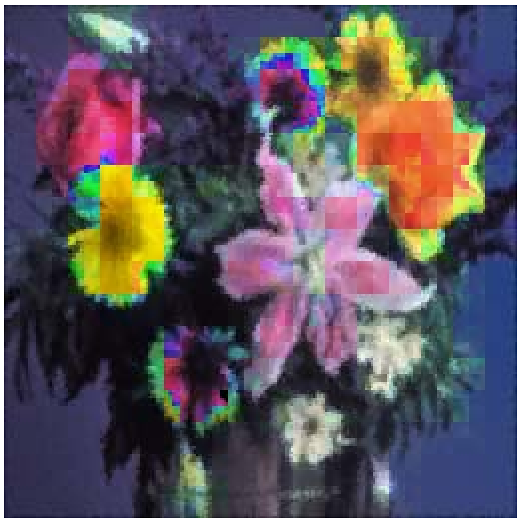
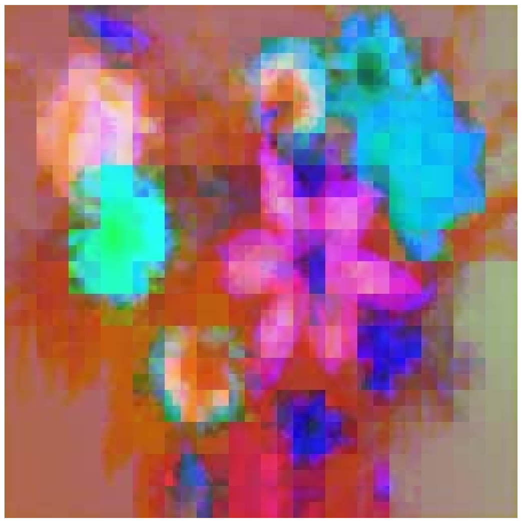
Size of the range blocks: H = 16, S = 16, and V = 4	
9) RGB Decompressed Image 	10) HSV Decompressed Image 

Figure 7.4: (Continued)

7.2 Compression ratio

Now, how to calculate the compression ratio? As an example, let us take the case of [H=4, S=128, V=8]:

1. Find the number of range blocks in each plane according to the given size of the range block.

   The number of the range blocks for H plane is: $(256 \times 256)/(4 \times 4) = 4096$ blocks.

   The number of the range blocks for S plane is: $(256 \times 256)/(128 \times 128) = 4$ blocks.

   The number of the range blocks for V plane is: $(256 \times 256)/(8 \times 8) = 1024$ blocks.

2. Find the number of bytes that's needed to store each fractal block. In our work it equals to 3 bytes, as discussed in chapter 4.

3. Find the number of bytes that's required to store the original image without compression. In our example it equals $(256 \times 256 \times 3) = 196608$ bytes.

4. Now the compression ratio is: $(4096 \times 3 + 4 \times 3 + 1024 \times 3) / 196608 = 13$.

As shown in table 7.2, most of the possible values of the compression ratio for different blocks size for H, S, and V planes respectively.

Table 7.2: Compression Ratio

Dimension of the range block			Compression Ratio	Dimension of the range block			Compression Ratio
H	S	V		H	S	V	
4	16	4	8:1	8	64	4	13:1
4	16	8	12:1	8	64	8	32:1
4	32	4	8:1	8	128	4	13:1
4	32	8	13:1	8	128	8	32:1
4	64	4	8:1	16	16	4	14:1
4	64	8	13:1	16	16	8	43:1
4	128	4	8:1	16	32	4	15:1
4	128	8	13:1	16	32	8	49:1
8	16	4	12:1	16	64	4	15:1
8	16	8	28:1	16	64	8	51:1
8	32	4	13:1	16	128	4	15:1
8	32	8	31:1	16	128	8	51:1

Applying the Three-Channel-HSV-Model produces images with high quality but with blocky bright areas, as shown in the left side of figure 7.5, this is due to the large blocks of the S plane. This can be solved easily by applying a 21×21 Relaxation filter (Average filter) to the S plane before transforming the HSV decompressed image to an RGB one, this is because human eyes act like a filter and take the mean of two colors. The results are shown in the right side of figure 7.5.

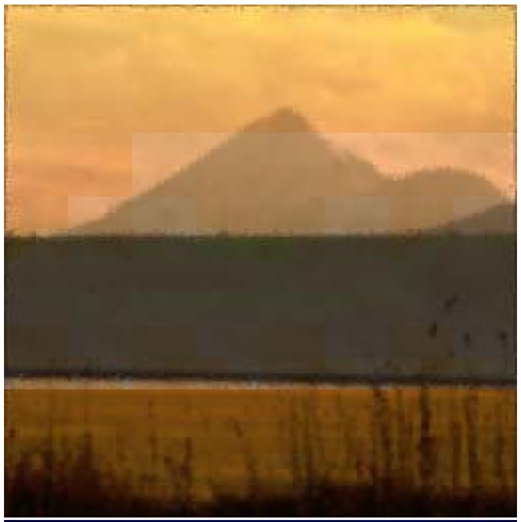
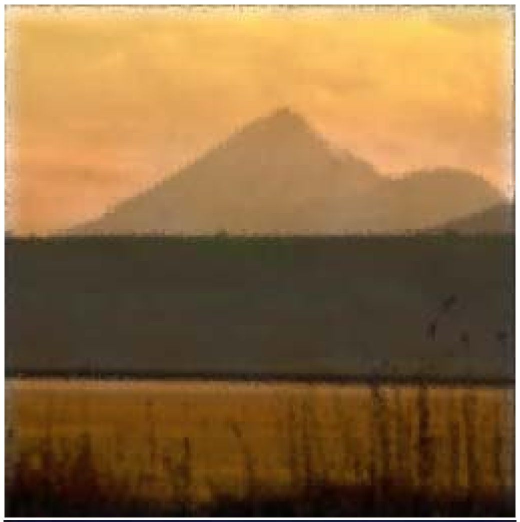
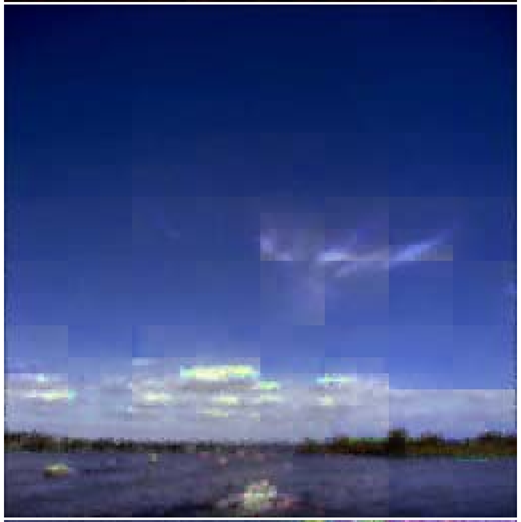
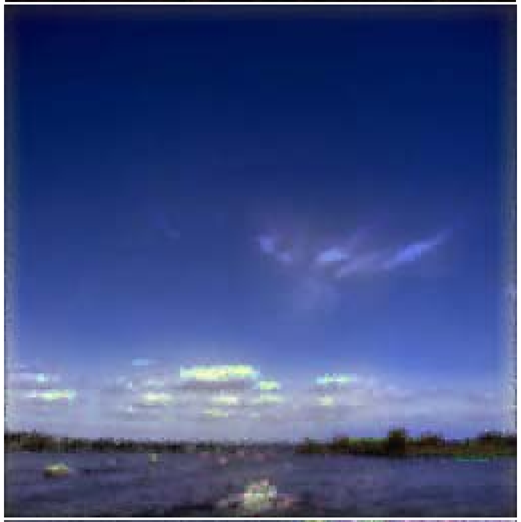
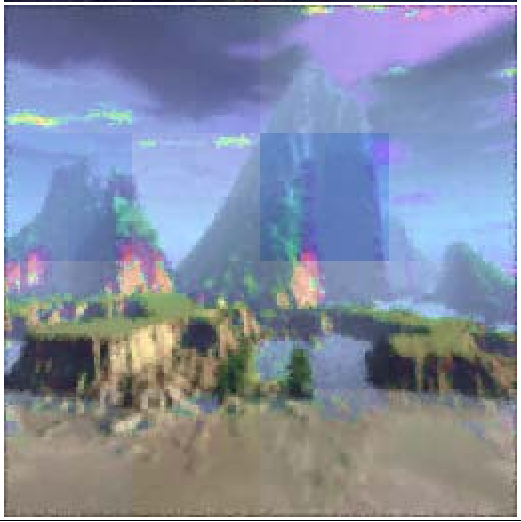
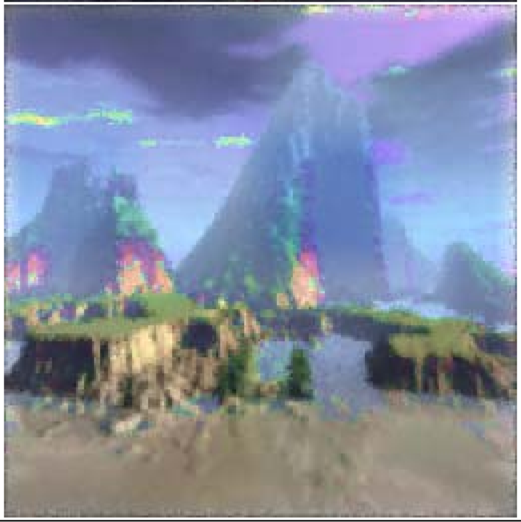
Images after applying Three-Channel-HSV-Model	Images after Applying a 21×21 Relaxation filter to the S plane
	
	
	

Figure 7.5: Applying a 21×21 Relaxation filter to the Decompressed Images

7.3 Comparison between Different Fractal Compression Methods

As shown in table 7.3, the Three-Channel-HSV-Model has the highest compression ratio. But in fact, we recommend the use of this model in images that have low variations in colors and little number of details. So, we prefer to choose one of the other models to encode high variations in colors and large number of details.

Table 7.3: Comparison between Fractal Compression Methods

Compression algorithm	Fractal code representation in bits	Compression Ratio
Three Channels RGB With 4×4 Range Blocks	$\left[\begin{array}{l|lllll} Red & i_R=6 & j_R=6 & w_R=3 & s_R=2 & o_R=6 \\ Green & i_G=6 & j_G=6 & w_G=3 & s_G=2 & o_G=6 \\ Blue & i_B=6 & j_B=6 & w_B=3 & s_B=2 & o_B=6 \end{array}\right]$	5:1
Master Plane With 4×4 Range Blocks	$i=6\ \ j=6\ \ w=3\ \ s_R=2\ \ s_G=2\ \ s_B=2\ \ o_R=6\ \ o_G=6\ \ o_B=6$	9:1
Vector Distortion With 4×4 Range Blocks	$i=6\ \ j=6\ \ w=3\ \ s_R=2\ \ s_G=2\ \ s_B=2\ \ o_R=6\ \ o_G=6\ \ o_B=6$	9:1
Hierarchal Coding With 4×4 Range Blocks	$i=6\ \ j=6\ \ w=3\ \ s_R=2\ \ s_G=2\ \ s_B=2\ \ o_R=6\ \ o_G=6\ \ o_B=6$	9:1
Three Channel-HSV-Model (in its worst case) With 4×4 Range Blocks for H &V, and 16×16 Range Block for S	$\left[\begin{array}{l|lllll} H & i_R=6 & j_R=6 & w_R=3 & s_R=2 & o_R=6 \\ S & i_G=4 & j_G=4 & w_G=3 & s_G=2 & o_G=6 \\ V & i_B=6 & j_B=6 & w_B=3 & s_B=2 & o_B=6 \end{array}\right]$	8:1
Three Channel-HSV-Model (in its best case) With 16×16 Range Block for H, 128×128 Range Block for S, and 8×8 Range Block for V	$\left[\begin{array}{l|lllll} H & i_R=4 & j_R=4 & w_R=3 & s_R=2 & o_R=6 \\ S & i_G=1 & j_G=1 & w_G=3 & s_G=2 & o_G=6 \\ V & i_B=5 & j_B=5 & w_B=3 & s_B=2 & o_B=6 \end{array}\right]$	51:1

As depicted in table 7.3, applying the three channels RGB model requires fractal encoding for each plane in the RGB image separately, this means that it needs to construct three fractal codes, one for each plane. Each fractal code composed of five components which are: the x and y positions, the type of the selected affine transformation, and the contrast and brightness values. So, encoding a (256×256×3) 8-bit image with range blocks of size 4×4, results a compression ratio of 5:1.

On the other hand, applying the master plane, the vector distortion, or the hierarchical coding algorithms, need one fractal code which contains the positions of x and y, the type of the selected affine transformation, and the contrast and brightness for the R, G and B respectively. So, encoding a (256×256×3) 8-bit image with range blocks of size 4×4, results a compression ratio of 9:1.

Finally, The Three-Channel-HSV-Model needs to construct three fractal codes, as depicted in table 7.3. Applying this model results a compression ratio of 8:1 in its worst case, but it can reach higher compression ratios as 51:1 or more, when it is applied to harmonic images.

CHAPTER EIGHT

CONCLUSIONS AND FUTURE WORK

At the beginning of my research about fractal image compression for colored images, I thought that we can reach the best model which is suitable for all types of images. But, after studying the previous models, we conclude that each model has a limited power which can make the model suitable in some situations. The implementation of the new model (Three-Channel-HSV-Model) achieves powerful results compared with previous models.

The first proposed idea was to reduce the three-plane colored image into one plane using the quantization technique; because we want to reduce the number of fractal encoders. So, that will reduce the encoding time. According to this idea, five models were proposed, therefore, the idea was applied firstly on the Reduced RGB model, in which we apply fractal compression over the reduced RGB image. But we found that not only it doesn't show good results but we need different bit representations for the reduced image depending on the image's colors, i.e. the dominant color in the image needs the highest number of bits. That's why we look for another color space which is the HSV.

In the next trials which were applied on the HSV color space, we discover that we don’t need to change the bit representation for the reduced image. For this reason, we proposed the Reduced HSV model that reduces the H, S and V planes into one plane. Note that applying this model shows a compression ratio of 16:1. But unfortunately, this doesn't imply good results. This is because, reducing the image -in this way- generates a lot of errors especially near edges, then applying fractal compression to the reduced image will produce another type of errors which results a degraded image.

After that, we attempt to reduce two planes into one plane, i.e. the reduced image will be composed of two planes only, and leaves the third one without quantization, so we need to apply fractal encoder twice, one for the reduced image and the other one for the remained plane. Depending on this idea, three models were implemented, which are the Reduced HS, Reduced SV and the Reduced HV models. Among those, the Reduced HV gives the worst results, since it tries to reduce the intensity and color planes in the image which have high information contents, while better results were achieved by applying the Reduced HS and SV models, where the reduced image combines planes with high information contents (H or V) and low information contents (S). So, we find that their results were good but their compression ratios were low (8:1).

After testing all the previous proposed models we conclude that reducing the image (either to one or two planes) is not a good idea; because it will produce level of errors. And by doing the reduction step, we will loose the biological intuition of the HSV model. Therefore, we must minimize the errors as possible, this will be by ignoring the reducing step in the previous models, this guides us, to fractal encode each HSV plane separately. At the first moment this looks very inefficient in terms of the number of fractal encoders, but applying fractal encoding three times and with different block sizes to each plane - large blocks for S and smaller ones to the H and V, depending on the properties of the image- will take less time. This model achieves a compression ratio of 51:1 or sometimes more than this in the case of highly harmonic images.

It is important to spot the light on the excellent results which are produced by applying this model, but these results may contain bright blocky areas that could be manipulated by sliding the relaxation filter over the S plane of the decompressed HSV image. Note that, we gain in experience that when we calculated the PSNR between the original and decompressed image, it will be an acceptable result if it is greater than or equals 50. Then in order to increase the efficiency of the model we apply the randomized approach which is discussed in chapter 4.

In summary, we recommend using the Three-Channel-HSV-Model for harmonic images that have low variations in colors and we can use other models like the Master Plane in the case of images that have a lot of details and high variations in colors, i.e. *the suitable model is in the suitable place*.

As a future work, we suggest to apply other types of partitioning methods such as the quadtree or horizontal vertical partitioning to improve the performance of the Three-Channel-HSV-Model and to build an intelligent system to determine the dominant color in the image.

REFERENCES

- Ali, Maaruf and Clarkson, G.Trevor. (1997). **Fractal Image Compression**, Proceeding on Information Technology and its Applications (ITA '91), Markfield Conference Center.

- Avıcbas, I. (2001). **Image Quality Statistics and their Use in Stegananalysis and Compression**. PhD Thesis, Bogasici University 2001.

- Banks, Stephen P. (1990). **Signal Processing, Image Processing and Pattern Recognition**. New York: Prentice Hall.

- Castleman, Kenneth. (1996). **Digital Image Processing**. New Jersey: Prentice Hall.

- De Oliverra, Jose F.L., Mandonca, Gelson V. and Dias, Roosevelt J. (1998). **A Modified Fractal Transformation to Improve the Quality of Fractal Coded Images**. ICIP (1) 1998: 756-759.

- Facolta di Scienze MM.FF .eNN. a Ca Vigual, Faculty of Mathematics. (1996). **Theory of Generalized Fractal Transforms**. Universita Degli Studi di Verona, Departement of Applied Mathematics, University of Waterloo.

- Fisher, Yuval. (1992). **Fractal Image Compression**, Siggraph 92 course Notes.

- Fisher, Yuval. (1994). **Fractal Image Compression Theory and Application**. New York: Springer-Verlag.

- Ghosh, S.K, Mukerjee, Jayanta and Das, P.P. (2003). **Fractal Image Compression: A Randomized Approach**. Elsevier, Pattern Recognition letters 25 (2004), 1013 – 1024.

- Gonzalez, Rafael C. (2001). **Digital Image Processing**, (2nd ed). New Jersey: Prentice Hall.

- Hearn, Donald and Baker, M. Pauline. (1997). **Computer Graphics**, (2nd ed). New Jersey: Prentice Hall

- Hürtgen, Bernd, Mols, Paul and Simon, Stephan F. (1994). **Fractal Transform Coding of Color Images**, Visual Communications and Image Processing, SPIE '94 (2308), 1683-1691.

- Jacquin, Arnaud E. (1991). **Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformations**. IEEE, Transactions on Image Processing, 1 (1): 18.

- Jähne, Bernd. (1997). **Practical Handbook on Image Processing for Scientific Applications**. Boca Raton, New York: CRC Press.

- Kominek, John, (1997). **Advances in Fractal Compression for Multimedia Applications**. Springer-Verlag, Systems 1997 (5): 255-270.

- Lewis, Rhys. (1990). **Practical Digital Image Processing**. New York: Ellis Horwood.

- Livingstone, MS and Hubel, DH. (1984). **Anatomy and physiology of a color system in the primate visual cortex**. J Neurosci (4):309–356.

- Mackinnon, Lachlan M., **Video and Image Compression Standards**. Multimedia Technology, F29IG2, January 2004.

- Nakano, Katsuhiko and Nakagawa, Masahiro. (2000). **A Hierarchical IFS Coding for Color Images**. Electronics and Communications in Japan, Part 3, 83(7): 1592-1599.

- Peitgen, Heinz-Otto, Jurgens, Hartmut and Saupe, Dietmar. (1992). **Fractals for the Classroom: Part One, Introduction to Fractals and Chaos**, (2nd ed). New York: Springer-Verlag.

- PINS, Markus and HILD, Hermann. (1989). **Variations on a Dither Algorithm.** Institut fur Betriebs und Dialogsysteme, University of Karlsruhe, West Germany.

- Puram, Venu-Gopal. (1999). **Image Coding Based on Fractal Theory of Iterated Contractive Image Transformations.**

- Sangwine, S.I. and Horne. R.E.N. (1998). **The colour Image Processing Handbook**, (1st ed). London: Chapman & Hall.

- Sobereano, Lisa A. (2000). **The Mathematical Foundation of Image Compression**. PhD Thesis. University of North Carolina. Wilmington, North Carolina.

- Texas Instruments Europe, **An Introduction to Fractal Image Compression**, Literature Number: BPRA065, October 1997.

- Turner, Martin J., Blackledge, Jonathan M. and Andrews, Patrick R. (1998). **Fractal Geometry in Digital Imaging**. San Diego: Academic Press.

- W.L.NG & H.A.Cohen, 10 June, 1994, "Theoretical Basis of Block- Oriented Fractal Image Encoding", Department of Computer Science and Computer Engineering La Trobe University.

- Welstead, Stephan. (1999). **Fractal and Wavelet Image Compression Techniques**. Washington USA: SPIE Optical Engineering Press.

- Wikipedia. (2001). **The Free Encyclopedia: HSV Color Space**.

- Wohlberg, B. and de Jager, G. (1999). **A Review of the Fractal Image Coding Literature**. IEEE, Transactions on Image Processing, 8 (12), 1716-1729

- Zhang, Ying and Po, Lie Man. (1995). **Fractal Color Image Compression using Vector Distortion Measure**. Proceedings ICIP-95 (IEEE International Conference on Image Processing), volume III, 284-287.

APPENDIX –A

Results of Applying Three-Channel-HSV-Model (Part I)

The results of applying the Three-Channel-HSV Model to nine (256×256, 8-bit) images:

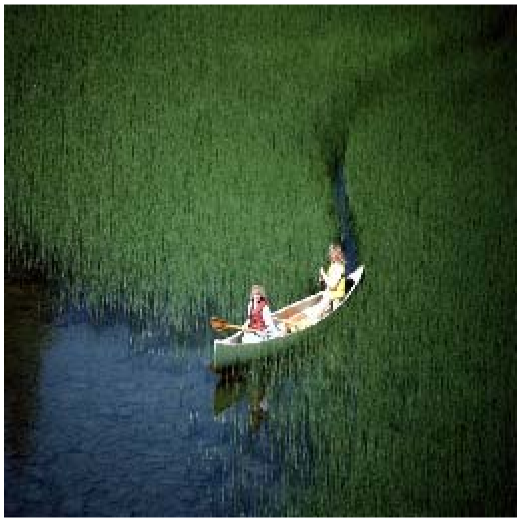
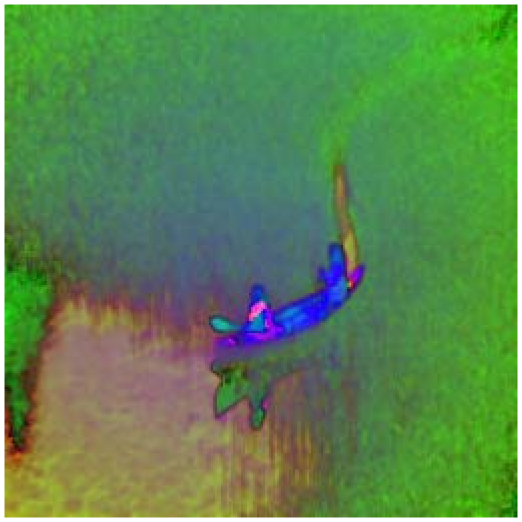
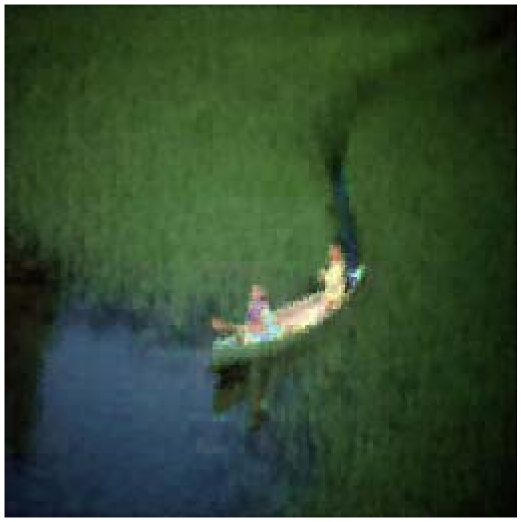
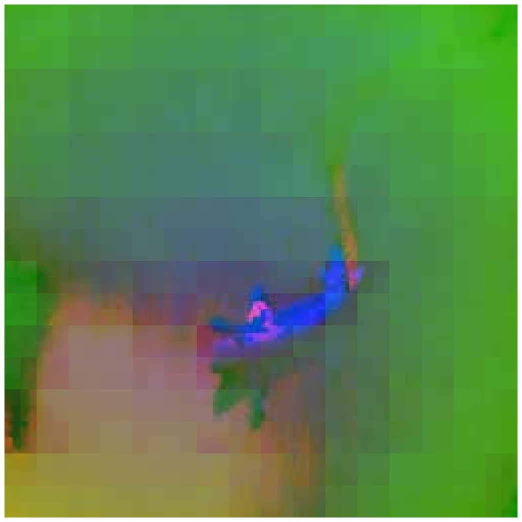
Image "a"	
1) Original RGB Image 	2) Original HSV Image 
Size of the range blocks: H =4, S = 32, and V = 4	
3) Decompressed RGB Image 	4) Decompressed HSV Image. 

Figure A.1: Applying Three-Channel-HSV-Model for Image "a"

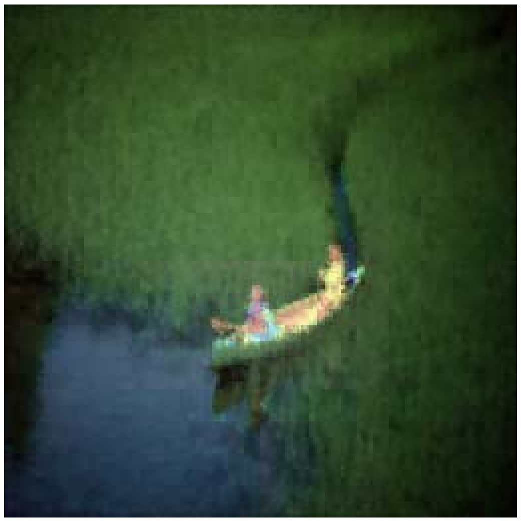
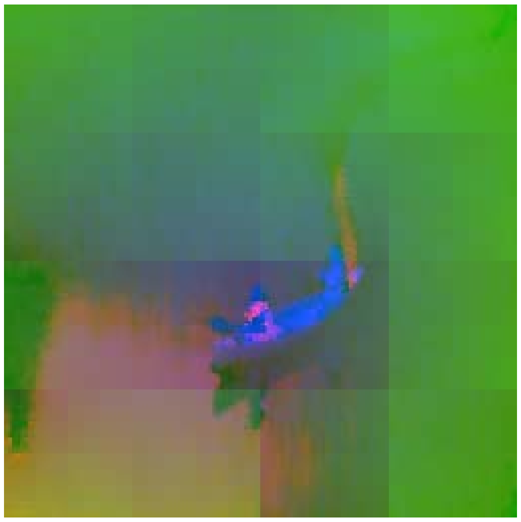
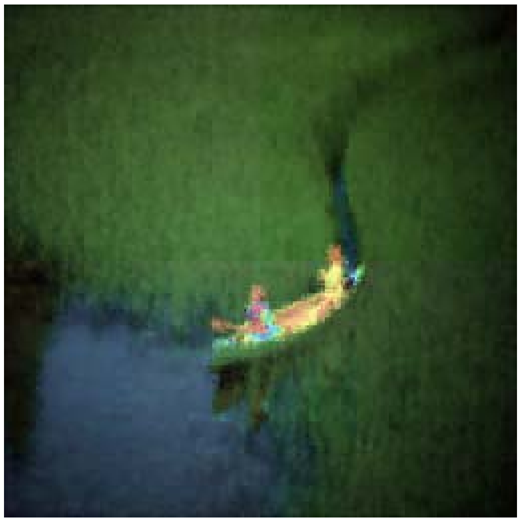
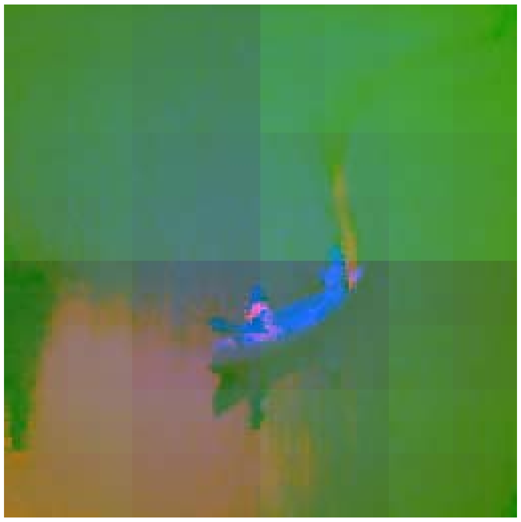
Size of the range blocks: H = 4, S = 64, and V = 4	
5) Decompressed RGB Image. 	6) Decompressed HSV Image. 
Size of the range blocks: H = 4, S = 128, and V = 4	
7) Decompressed RGB Image. 	8) Decompressed HSV Image. 

Figure A.1: (Continued)

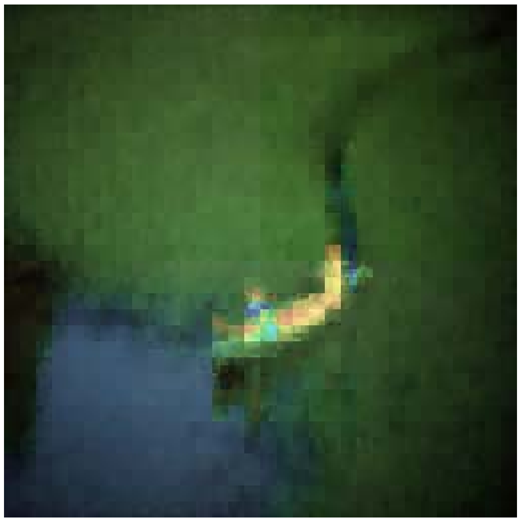
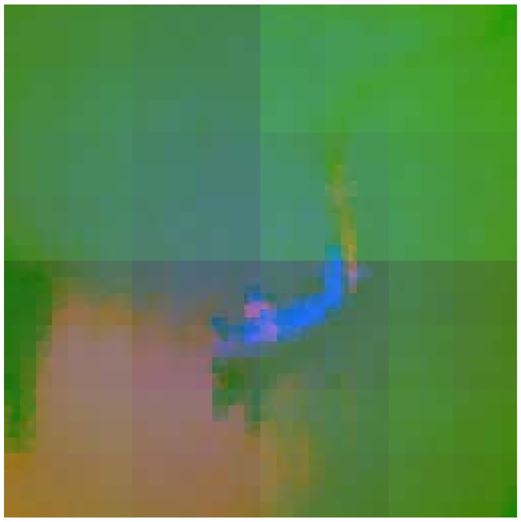
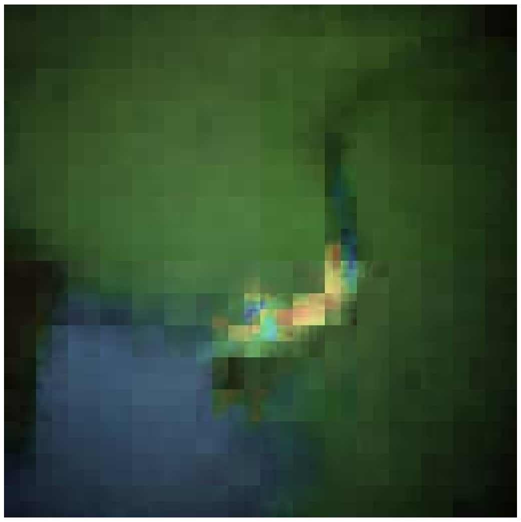
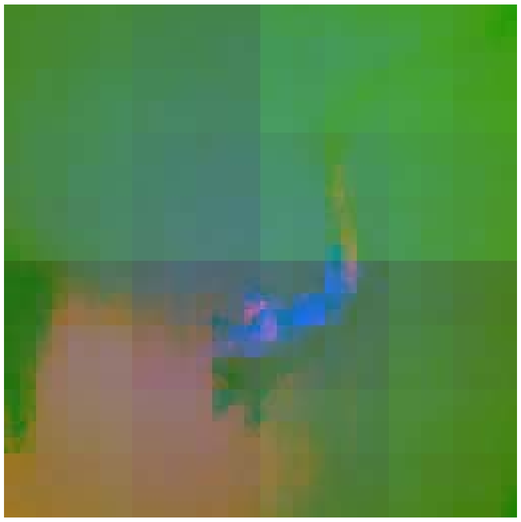
Size of the range blocks: H = 8, S = 128, and V = 8	
9) Decompressed RGB Image. 	10) Decompressed HSV Image. 
Size of the range blocks: H = 8, S = 128, and V = 16	
11) Decompressed RGB Image. 	12) Decompressed HSV Image. 

Figure A.1: (Continued)

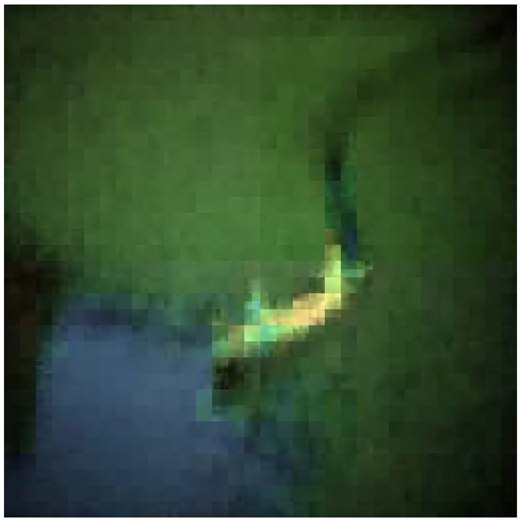
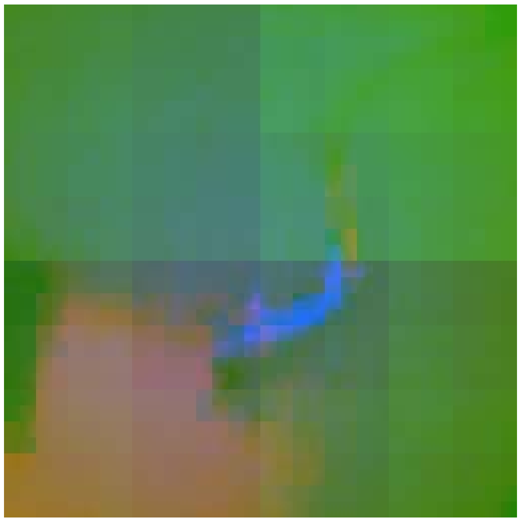
Size of the range blocks: H = 16, S = 128, and V = 8	
13) Decompressed RGB Image. 	14) Decompressed HSV Image. 

Figure A.1: (Continued)

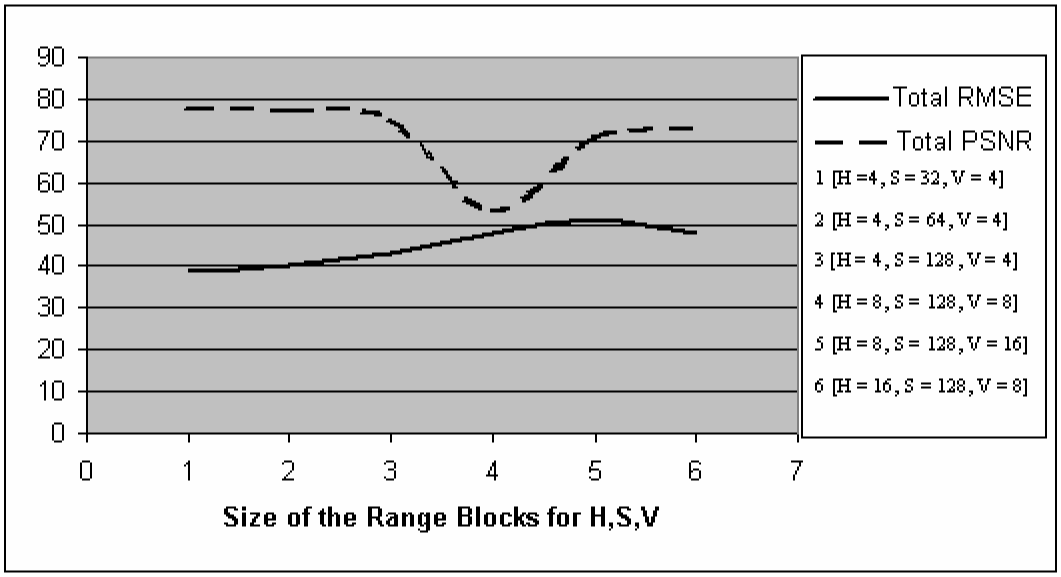


Figure A.2: RMSE and PSNR values for Image "a"

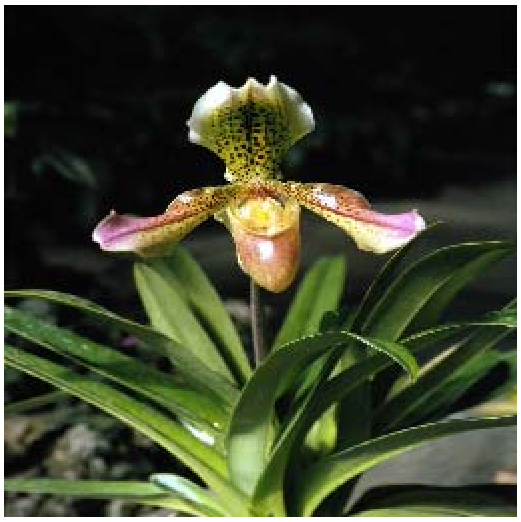
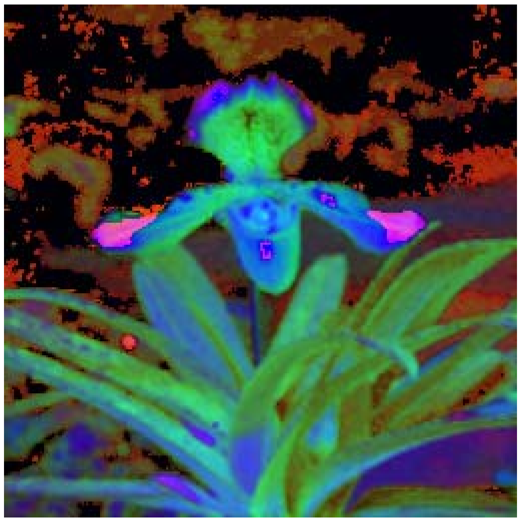
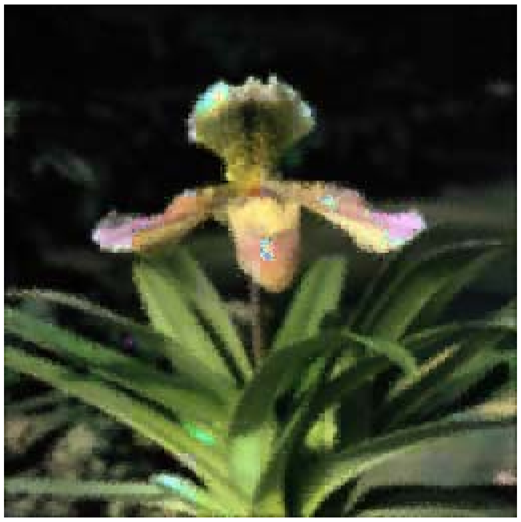
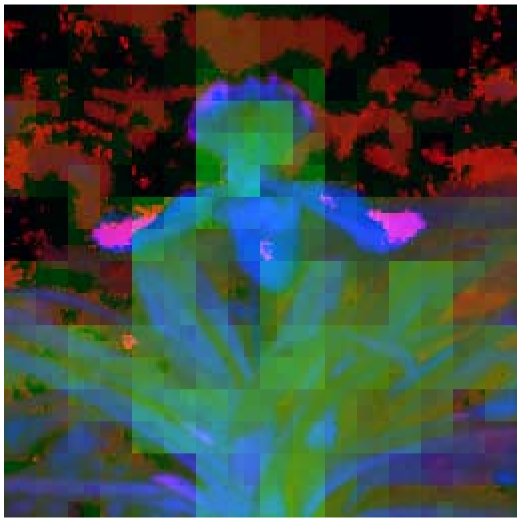
Image "b"	
1) Original RGB Image. 	2) Original HSV Image. 
Size of the range blocks: H = 4, S = 32, and V = 4	
3) Decompressed RGB Image. 	4) Decompressed HSV Image. 

Figure A.3: Applying Three-Channel-HSV-Model to Image "b"

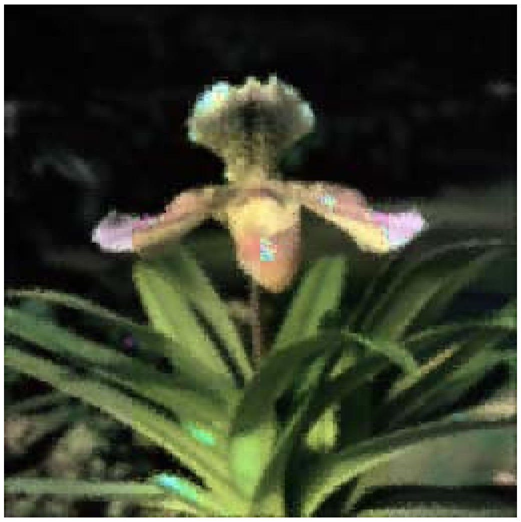
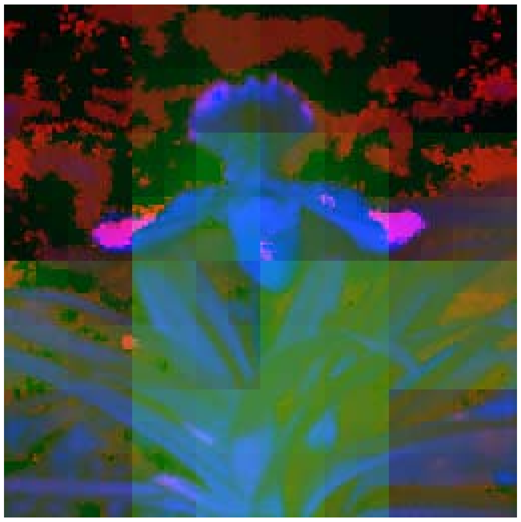
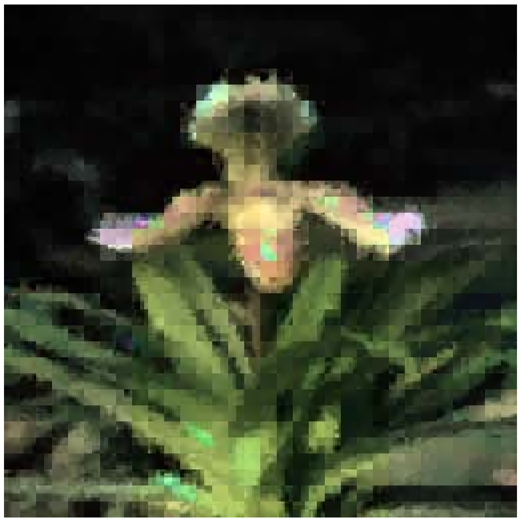
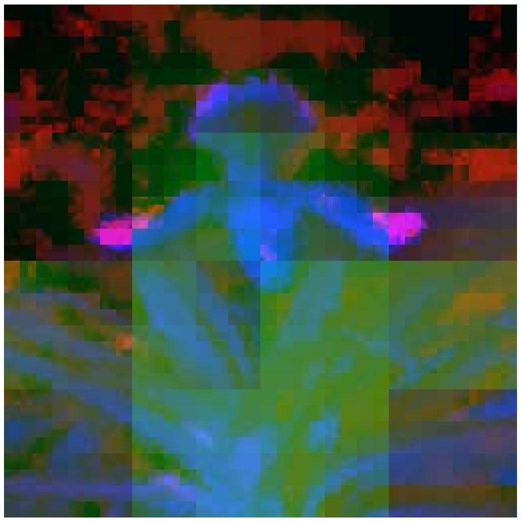
Size of the range blocks: H = 4, S = 64, and V = 4	
5) Decompressed RGB Image. 	6) Decompressed HSV Image. 
Size of the range blocks: H = 8, S = 64, and V = 8	
7) Decompressed RGB Image. 	8) Decompressed HSV Image. 

Figure A.3: (Continued)

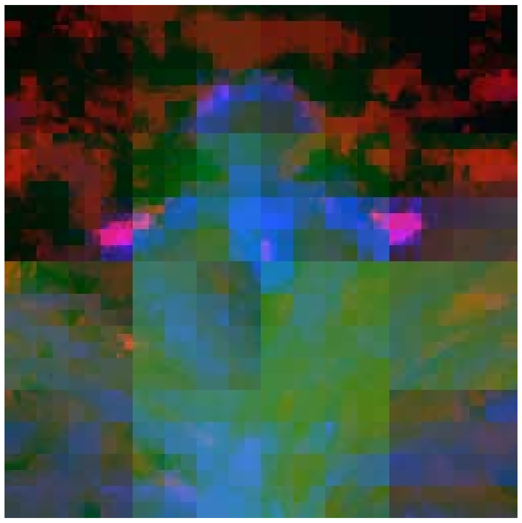
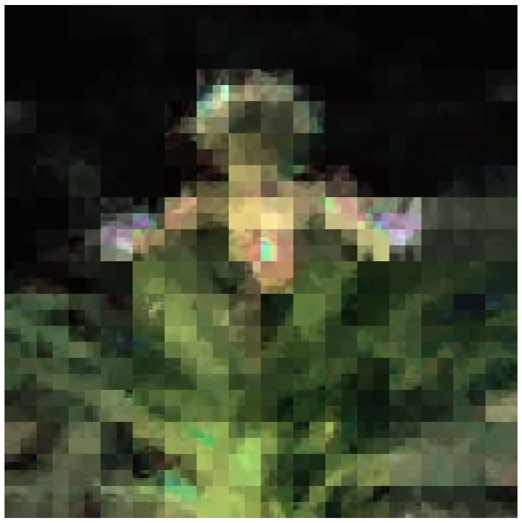
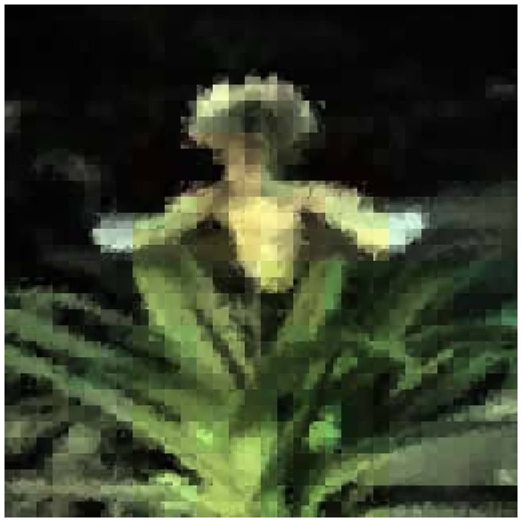
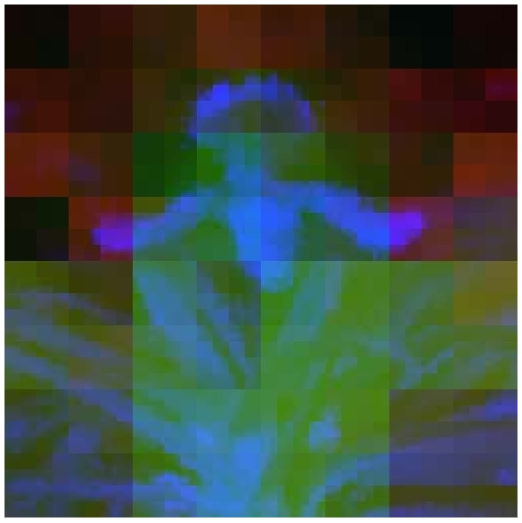
Size of the range blocks: H = 8, S = 64, and V = 16	
9) Decompressed RGB Image. 	10) Decompressed HSV Image 
Size of the range blocks: H = 32, S = 64, and V = 8	
11) Decompressed RGB Image 	12) Decompressed HSV Image 

Figure A.3: (Continued)

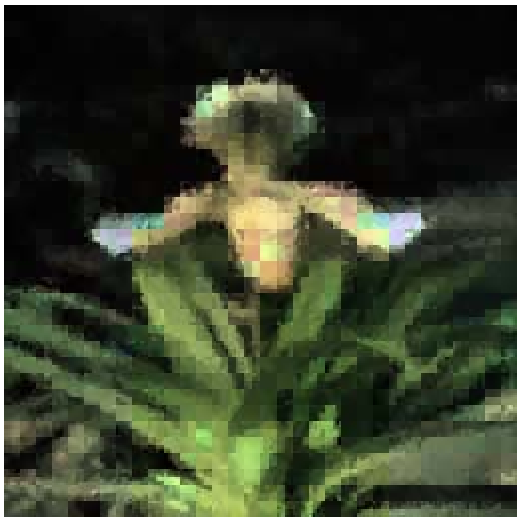
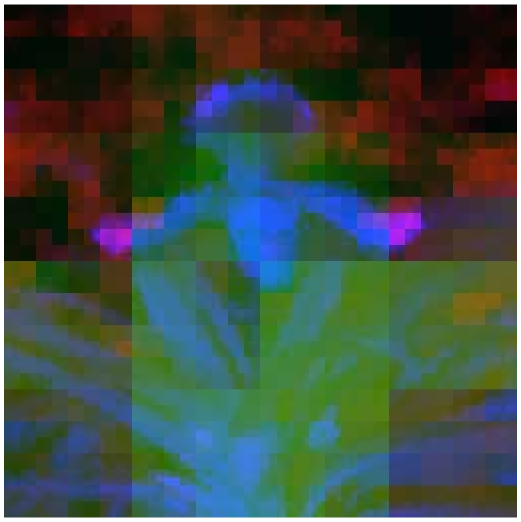
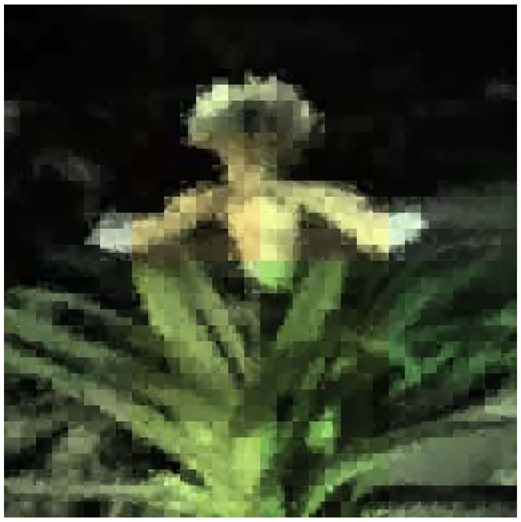
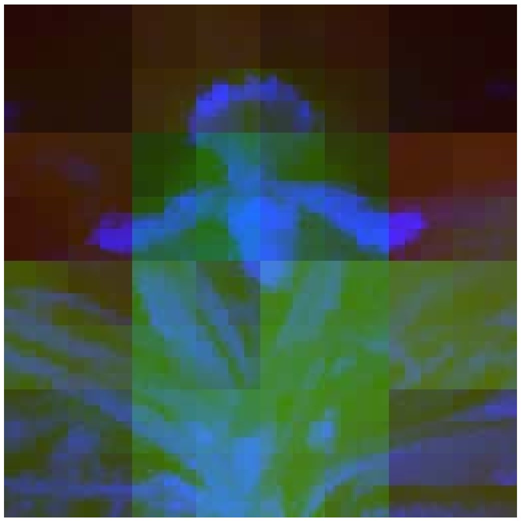
Size of the range blocks: H = 16, S = 64, and V = 8	
13) Decompressed RGB Image 	14) Decompressed HSV Image 
Size of the range blocks: H = 64, S = 64, and V = 8	
15) Decompressed RGB Image. 	16) Decompressed HSV Image. 

Figure A.3: (Continued)

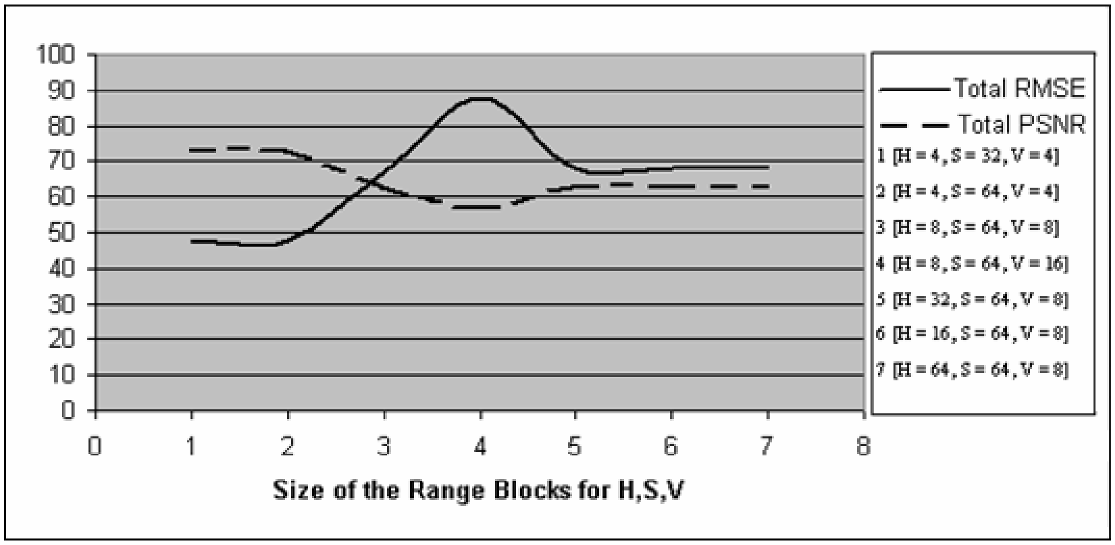


Figure A.4: RMSE and PSNR values for Image "b"

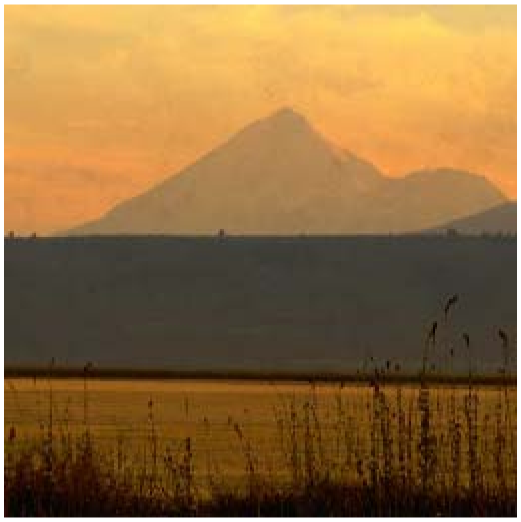
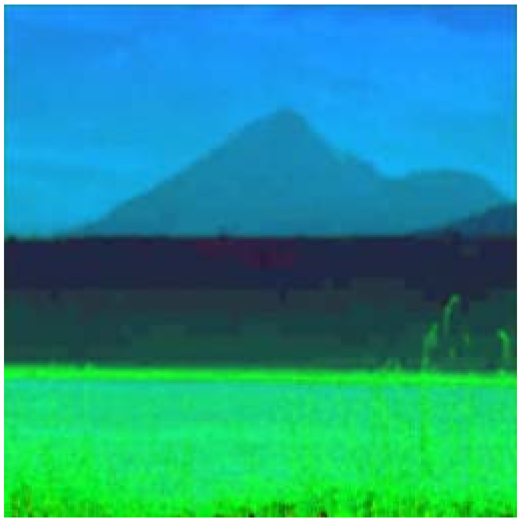
Image "c"	
1) Original RGB Image. 	2) Original HSV Image. 

Figure A.5: Applying Three-Channel-HSV-Model to Image "c"

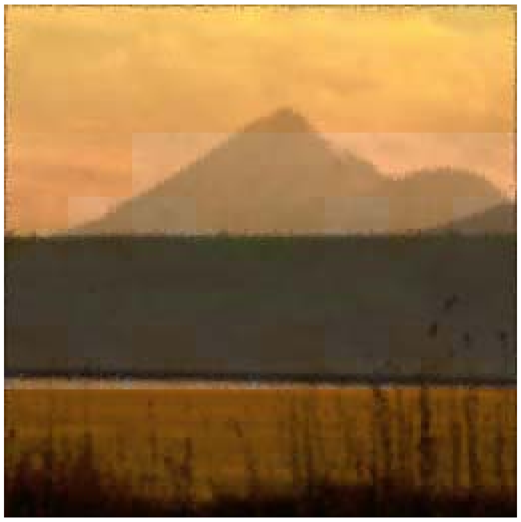
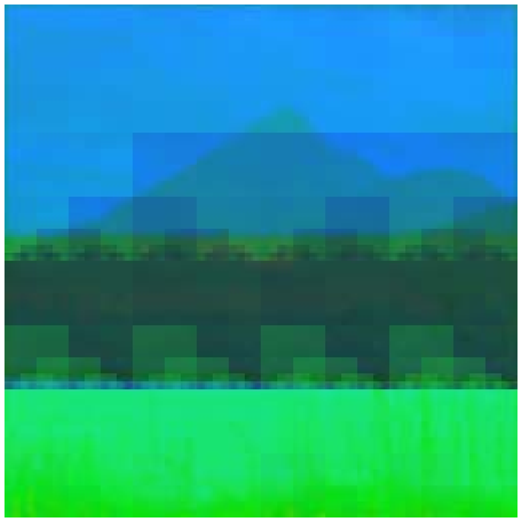
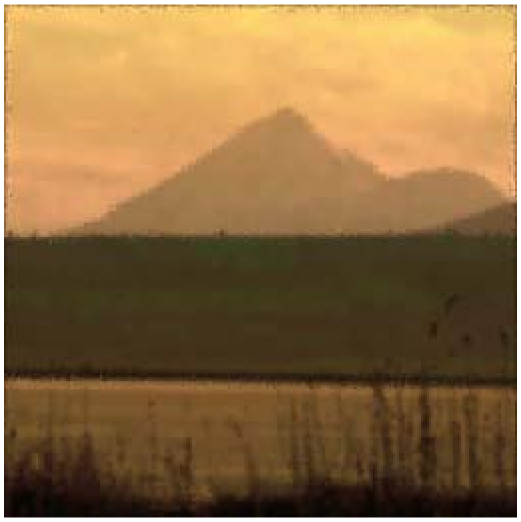
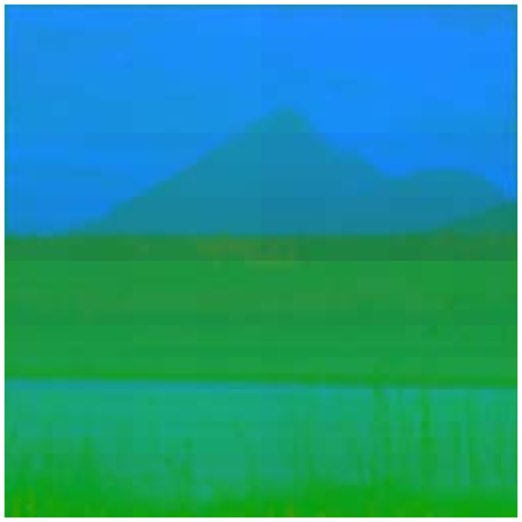
Size of the range blocks: H = 4, S = 64, and V = 4	
3) Decompressed RGB Image. 	4) Decompressed HSV Image. 
Size of the range blocks: H = 4, S = 128, and V = 4	
5) Decompressed RGB Image. 	6) Decompressed HSV Image. 

Figure A.5: (Continued)

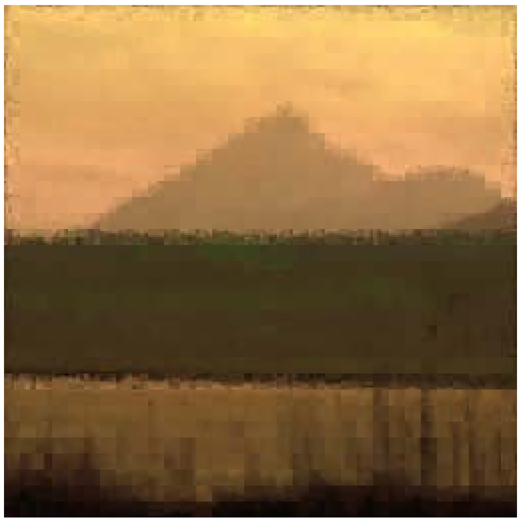
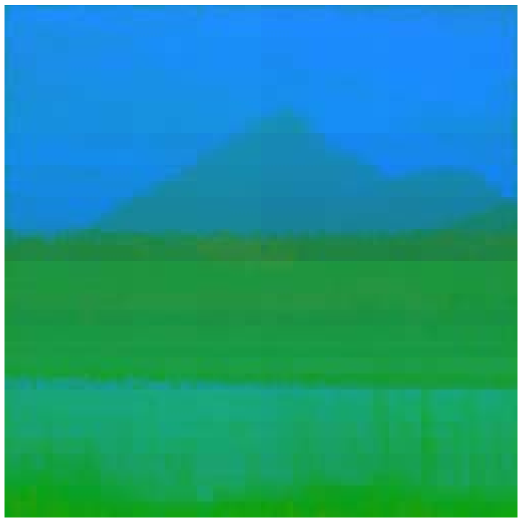
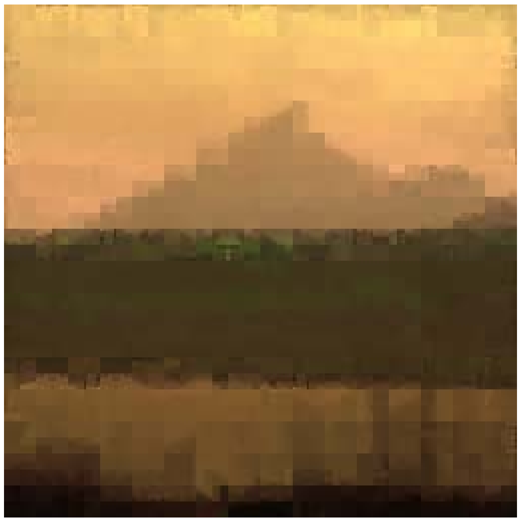
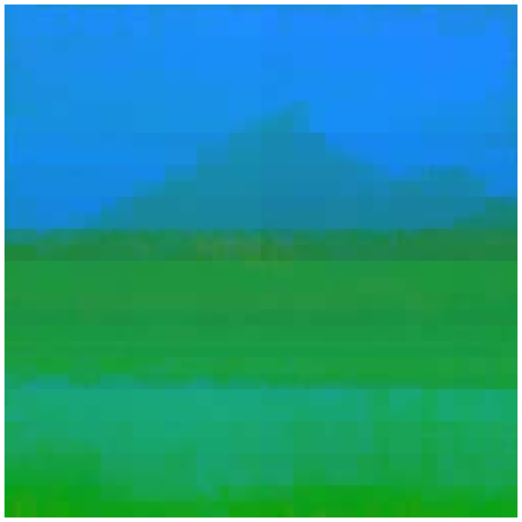
Size of the range blocks: H = 8, S = 128, and V = 8	
7) Decompressed RGB Image. 	8) Decompressed HSV Image. 
Size of the range blocks: H = 8, S = 128, and V = 16	
9) Decompressed RGB Image. 	10) Decompressed HSV Image. 

Figure A.5: (Continued)

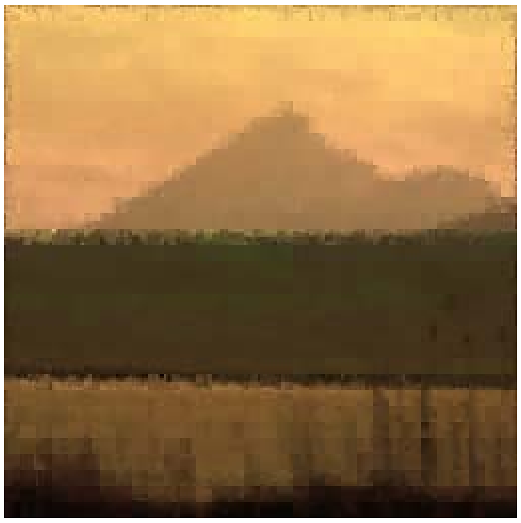
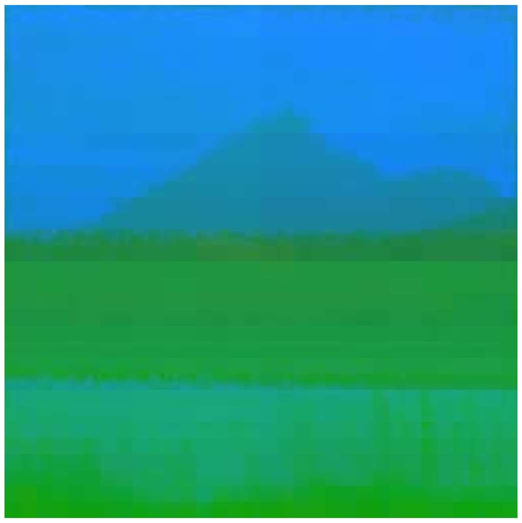
Size of the range blocks: H = 16, S = 128, and V = 8	
11) Decompressed RGB Image. 	12) Decompressed HSV Image. 

Figure A.5: (Continued)

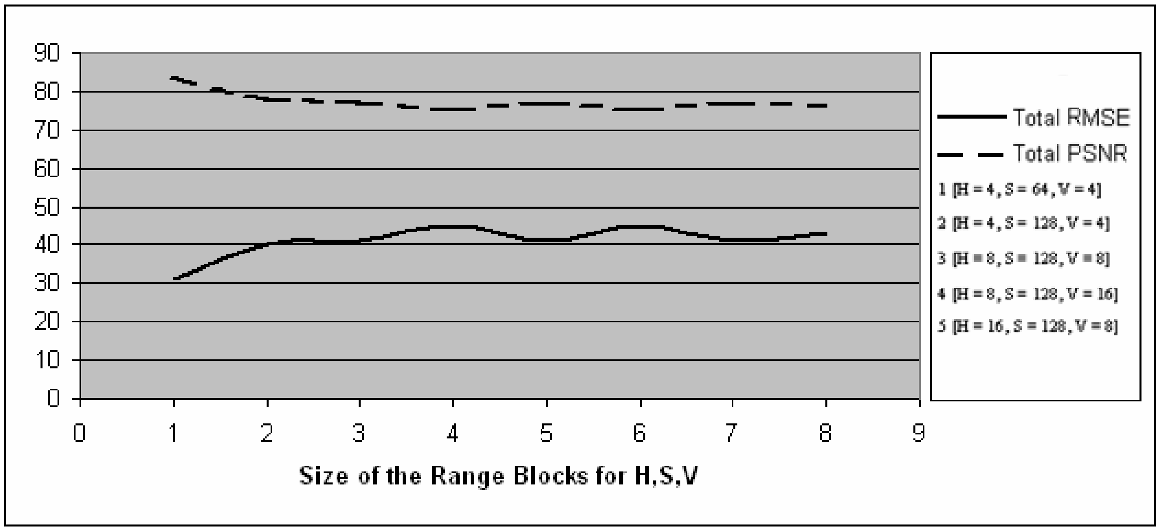


Figure A.6: RMSE and PSNR values for Image "c"

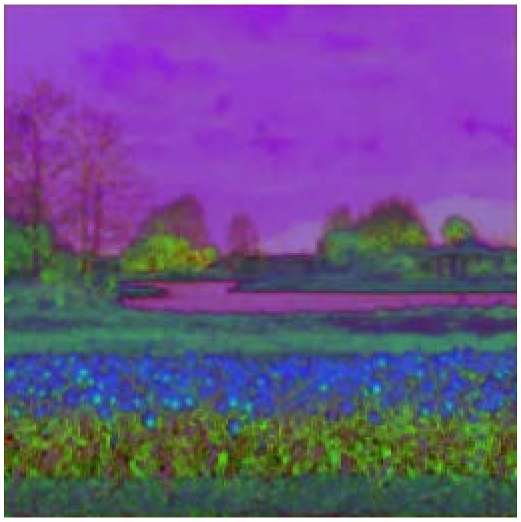
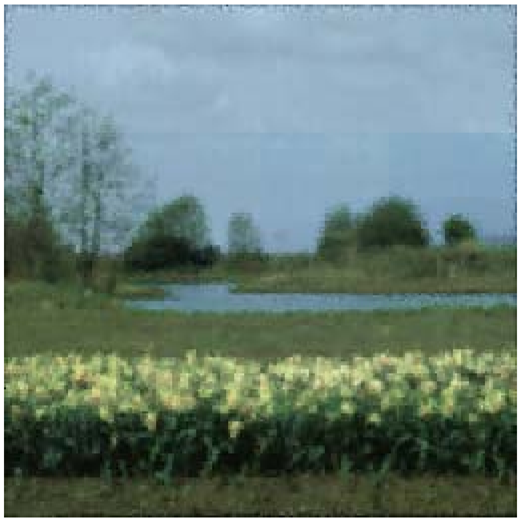
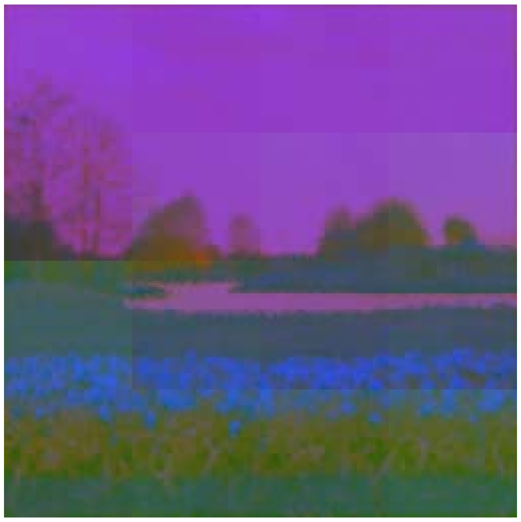
Image "d"	
1) Original RGB Image.	2) Original HSV Image. 
Size of the range blocks: H = 4, S = 64, and V = 4	
3) Decompressed RGB Image. 	4) Decompressed HSV Image. 

Figure A.7: Applying Three-Channel-HSV-Model to Image "d"

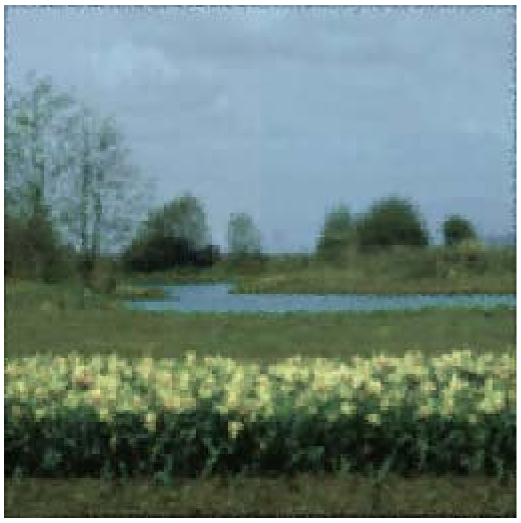
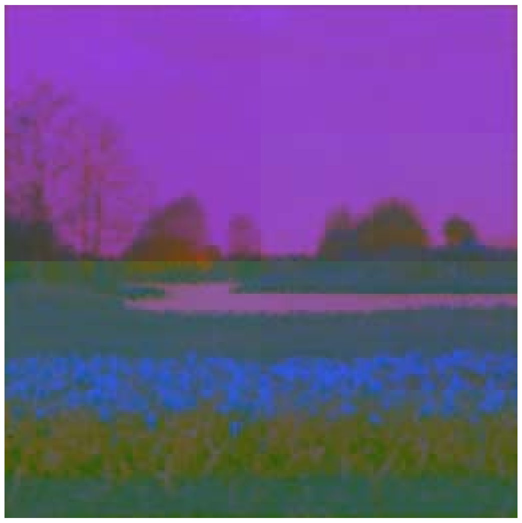
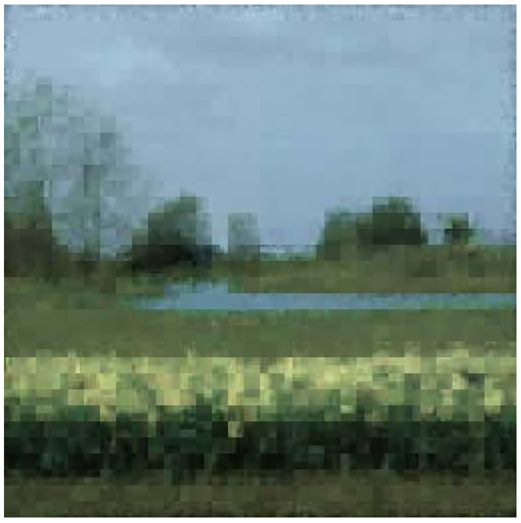
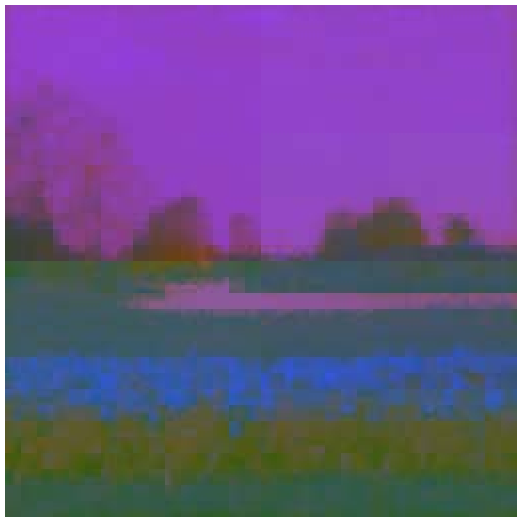
Size of the range blocks: H = 4, S = 128, and V = 4	
5) Decompressed RGB Image. 	6) Decompressed HSV Image. 
Size of the range blocks: H = 8, S = 128, and V = 8	
7) Decompressed RGB Image. 	8) Decompressed HSV Image. 

Figure A.7: (Continued)

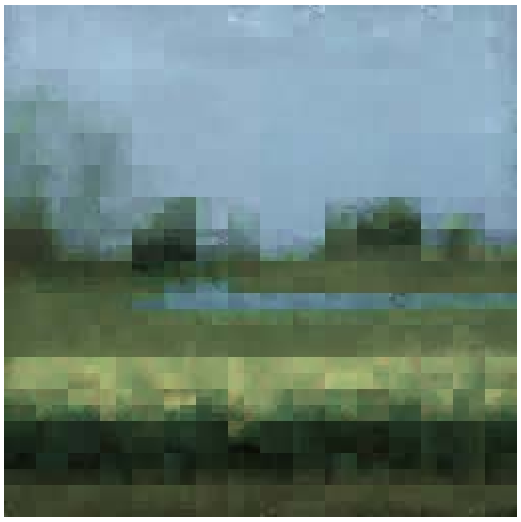
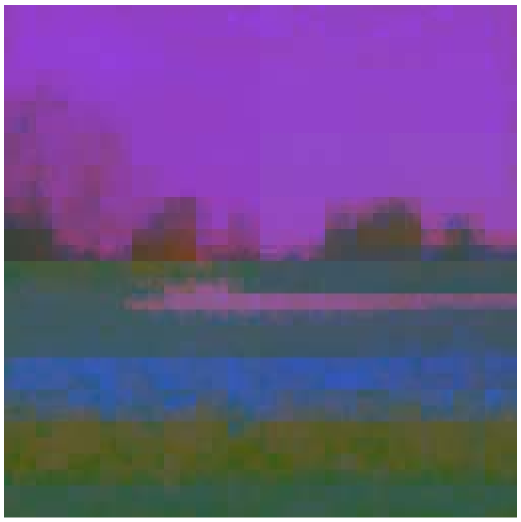
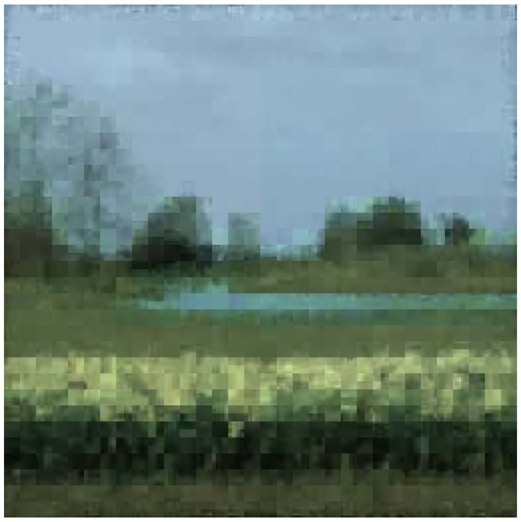
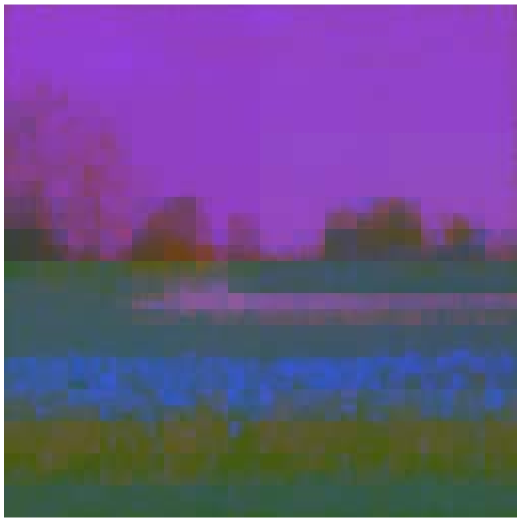
Size of the range blocks: H = 8, S = 128, and V = 16	
9) Decompressed RGB Image. 	10) Decompressed HSV Image. 
Size of the range blocks: H = 16, S = 128, and V = 8	
11) Decompressed RGB Image. 	12) Decompressed HSV Image. 

Figure A.7: (Continued)

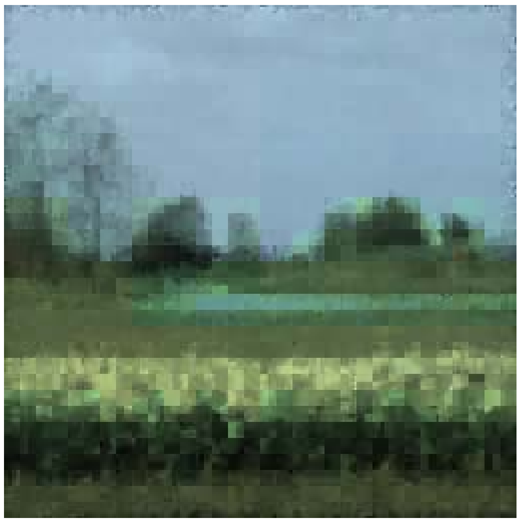
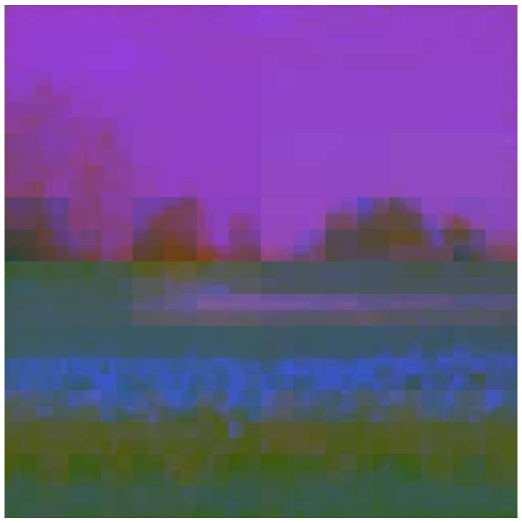
Size of the range blocks: H = 32, S = 128, and V = 8	
13) Decompressed RGB Image. 	14) Decompressed HSV Image. 

Figure A.7: (Continued)

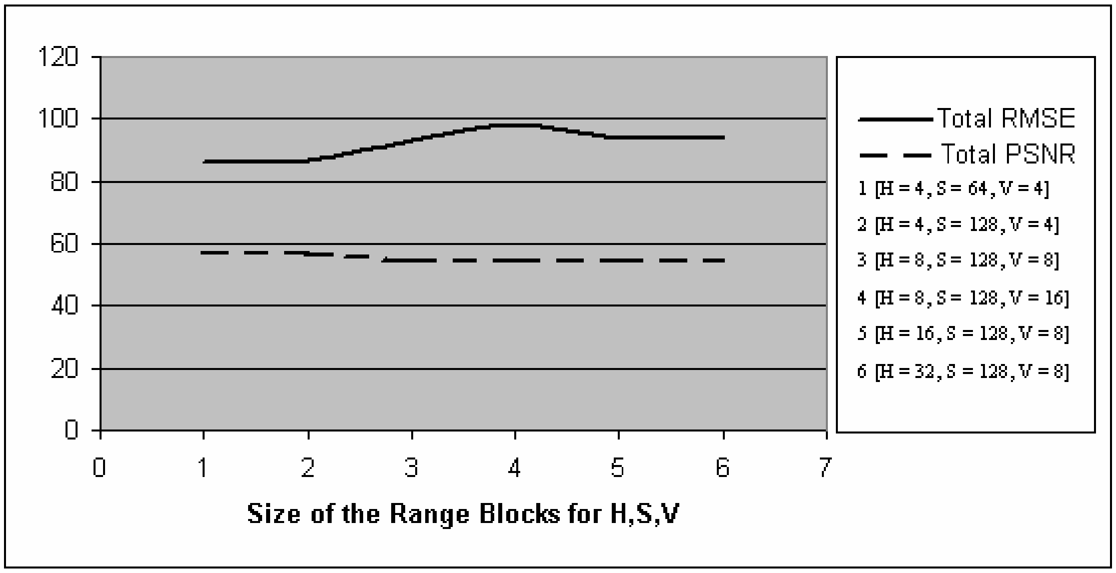


Figure A.8: RMSE and PSNR values for Image "d"

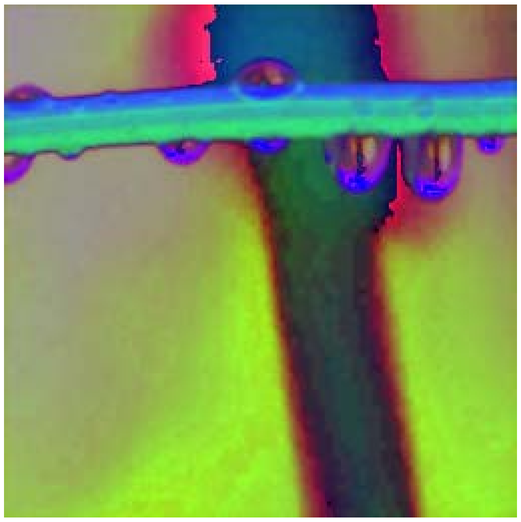
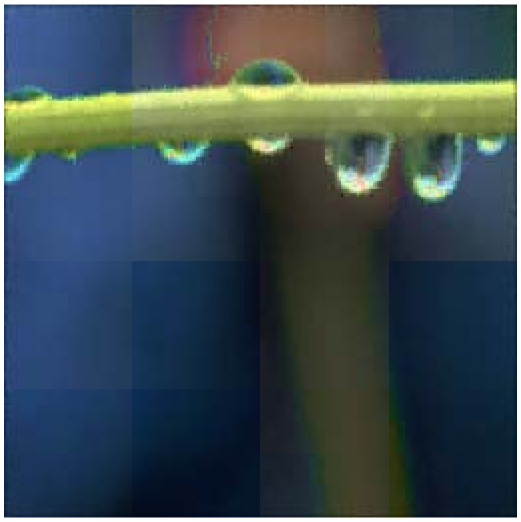
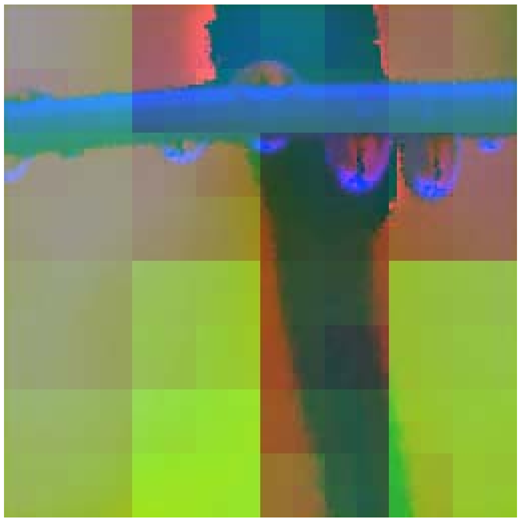
Image "e"	
1) Original RGB Image.	2) Original HSV Image. 
Size of the range blocks: H = 4, S = 64, and V = 4	
3) Decompressed RGB Image. 	4) Decompressed HSV Image. 

Figure A.9: Applying Three-Channel-HSV-Model to Image "e"

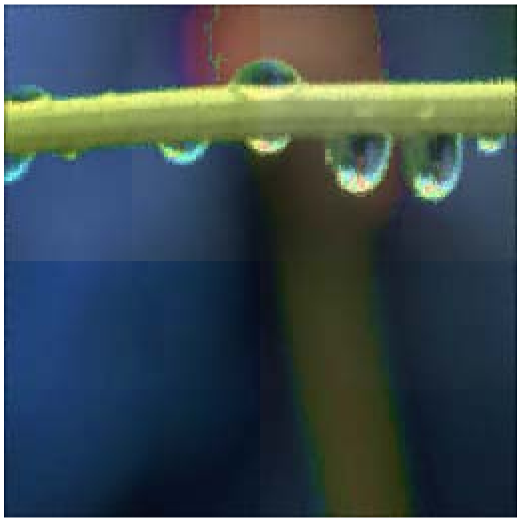
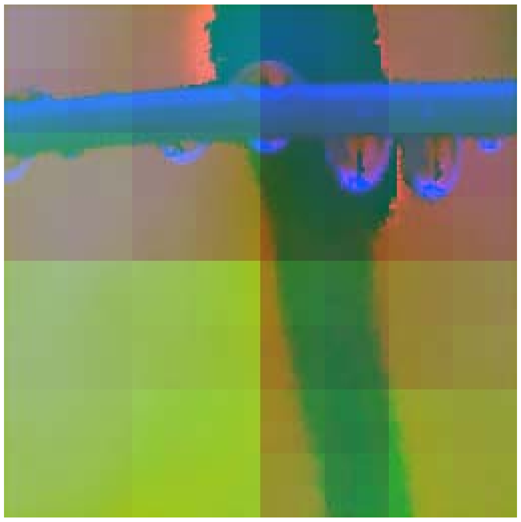
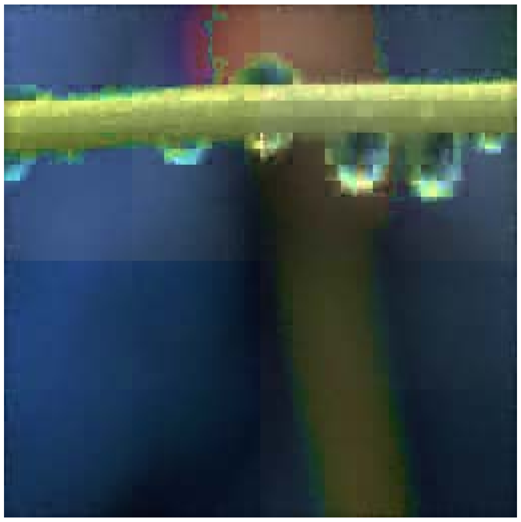
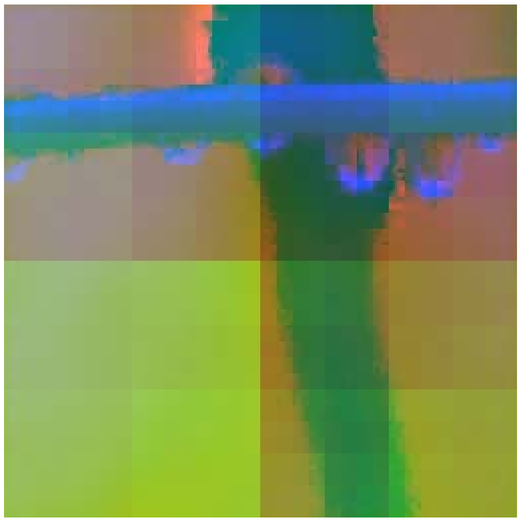
Size of the range blocks: H = 4, S = 128, and V = 4	
5) Decompressed RGB Image. 	6) Decompressed HSV Image. 
Size of the range blocks: H = 8, S = 128, and V = 8	
7) Decompressed RGB Image. 	8) Decompressed HSV Image. 

Figure A.9: (Continued)

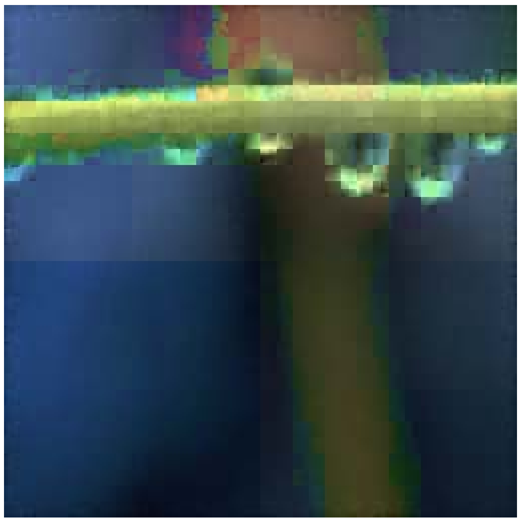
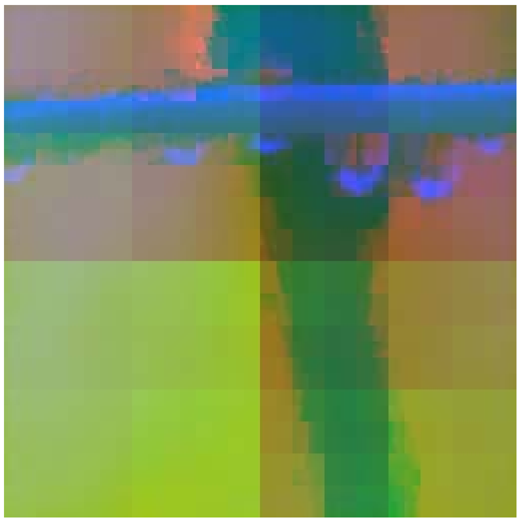
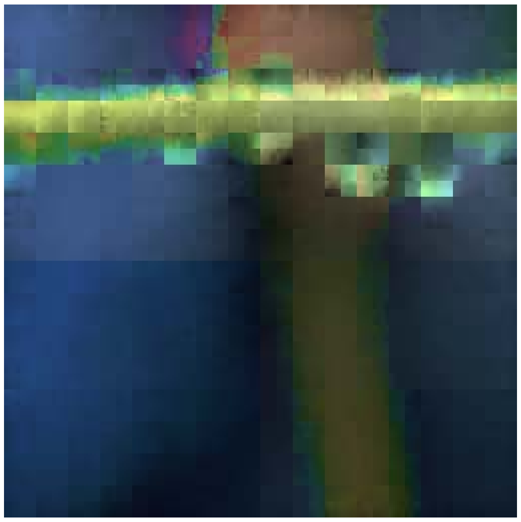
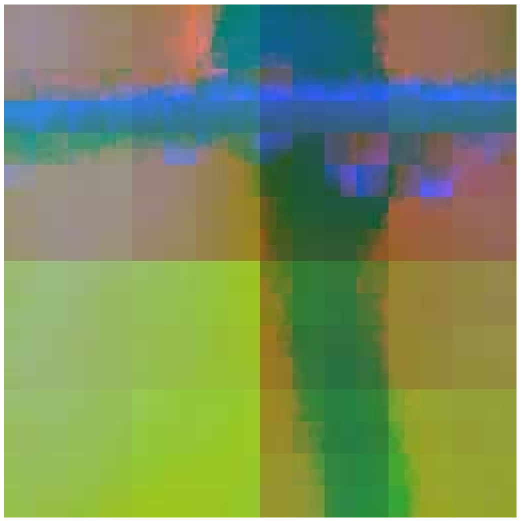
Size of the range blocks: H = 16, S = 128, and V = 8	
9) Decompressed RGB Image. 	10) Decompressed HSV Image. 
Size of the range blocks: H = 16, S = 128, and V = 16	
11) Decompressed RGB Image. 	12) Decompressed HSV Image. 

Figure A.9: (Continued)

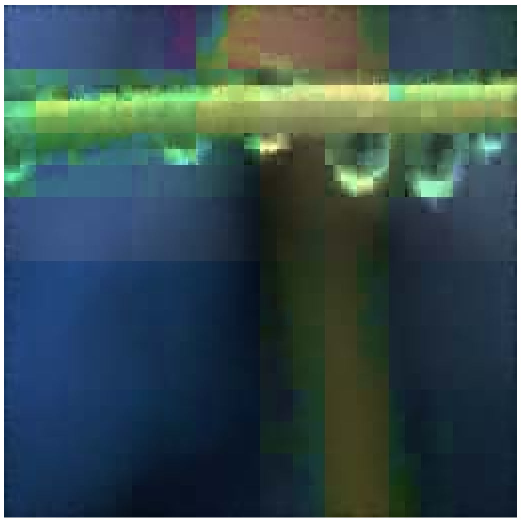
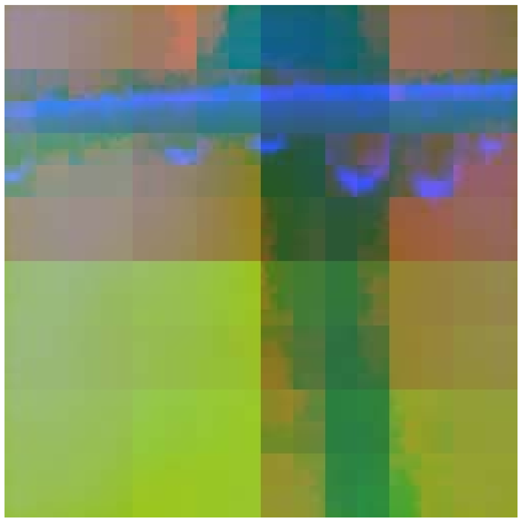
Size of the range blocks: H = 32, S = 128, and V = 8	
13) Decompressed RGB Image. 	14) Decompressed HSV Image. 

Figure A.9: (Continued)

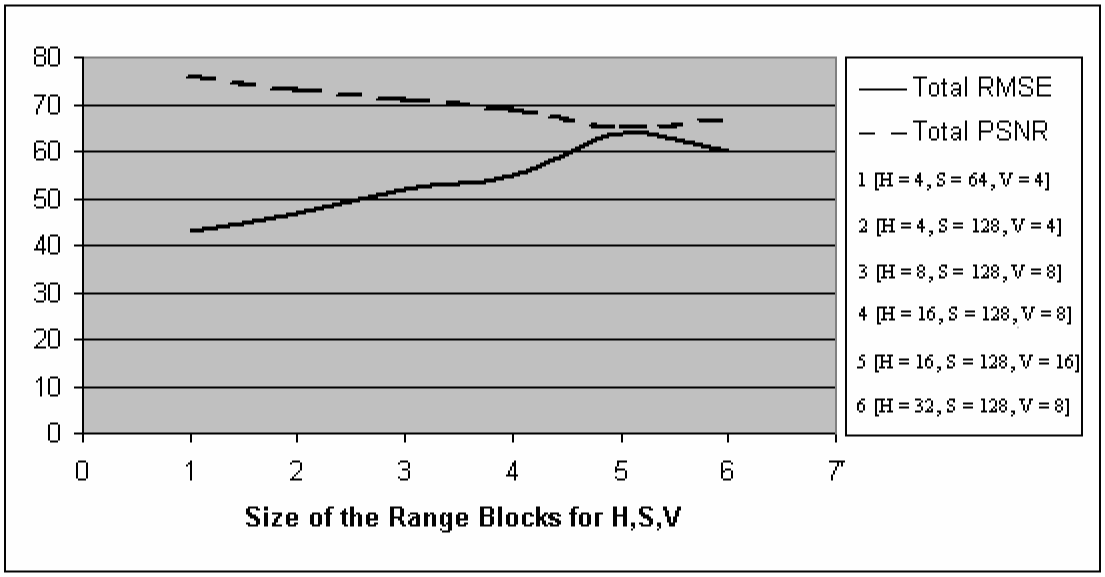


Figure A.10: RMSE and PSNR values for Image "e"

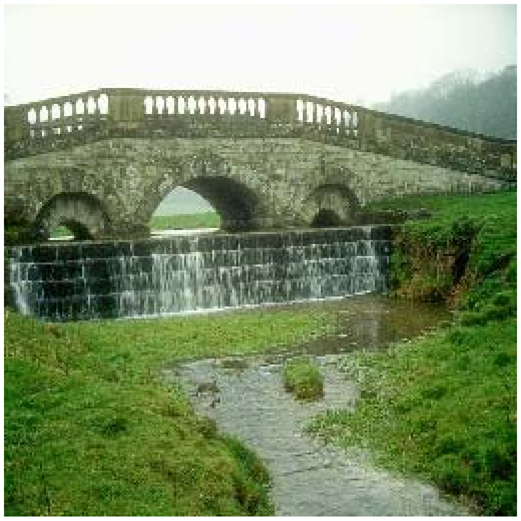
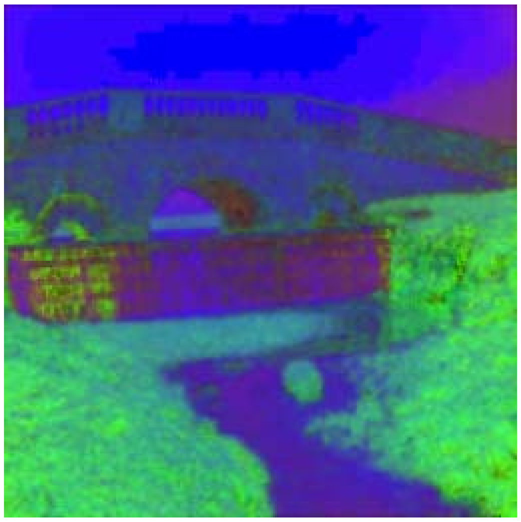
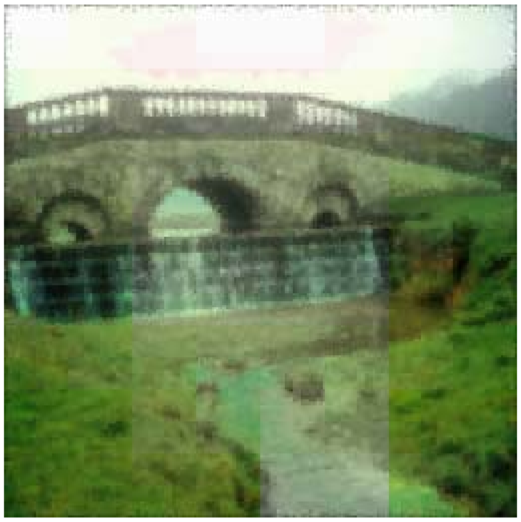
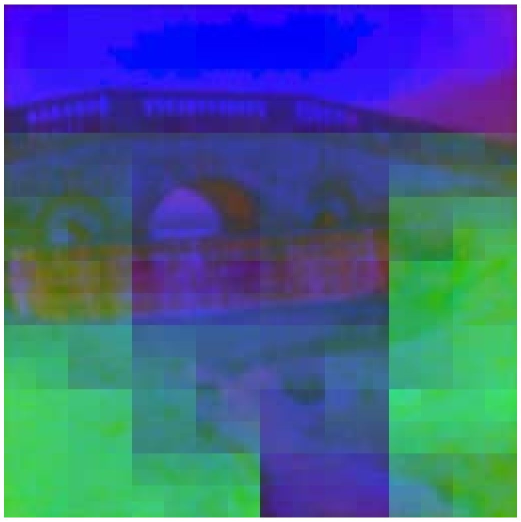
Image "f"	
1) Original RGB Image. 	2) Original HSV Image. 
Size of the range blocks: H = 4, S = 64, and V = 4	
3) Decompressed RGB Image. 	4) Decompressed HSV Image. 

Figure A.11: Applying Three-Channel-HSV-Model to Image "f"

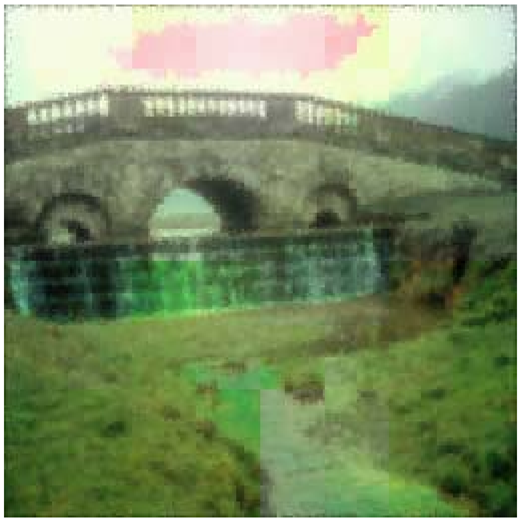
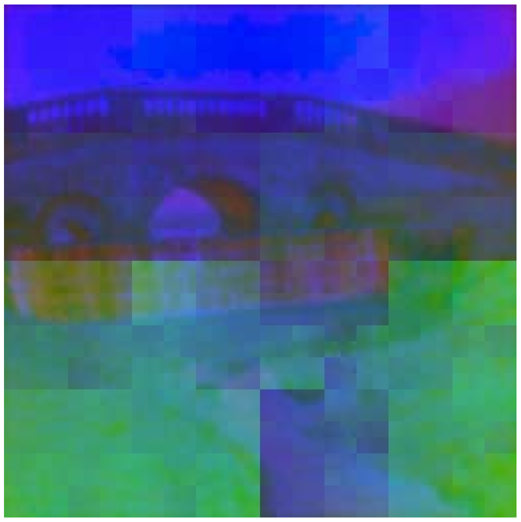
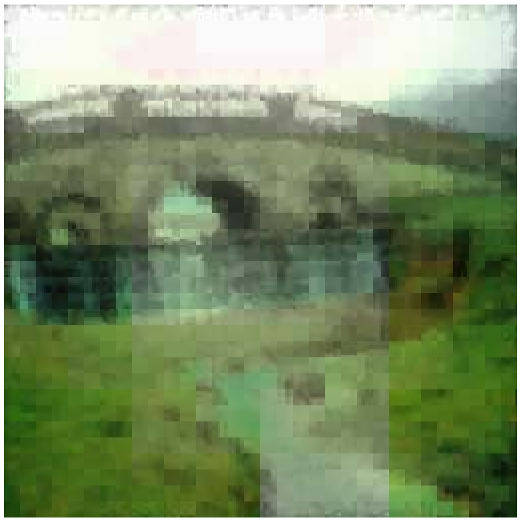
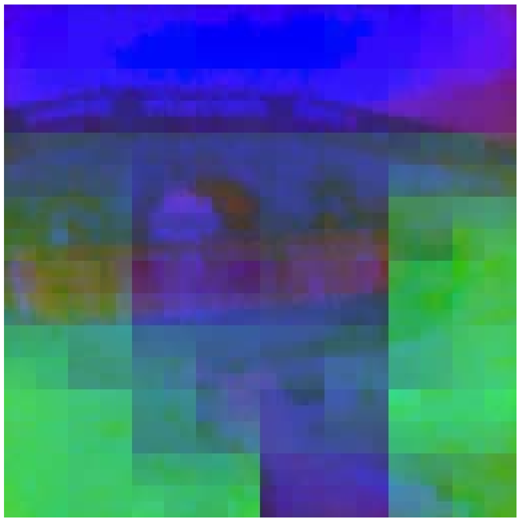
Size of the range blocks: H = 4, S = 128, and V = 4	
5) Decompressed RGB Image. 	6) Decompressed HSV Image. 
Size of the range blocks: H = 8, S = 64, and V = 8	
7) Decompressed RGB Image. 	8) Decompressed HSV Image. 

Figure A.11: (Continued)

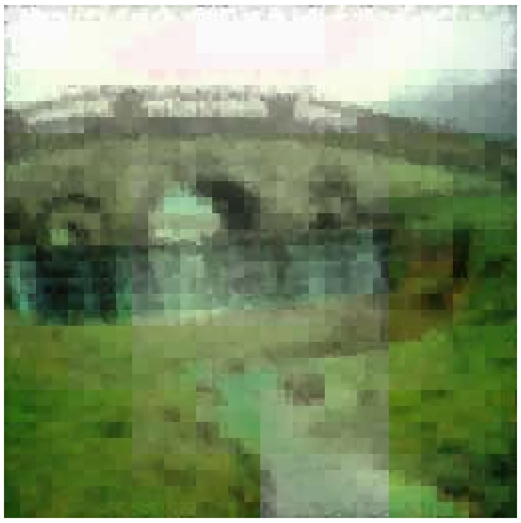
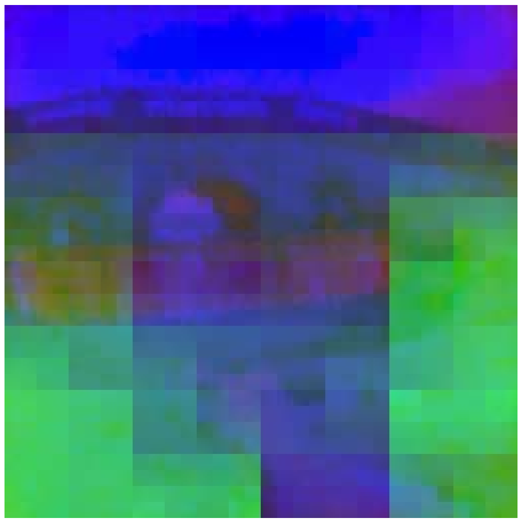
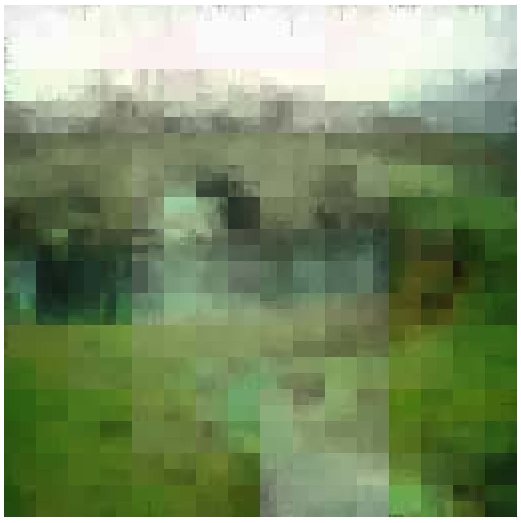
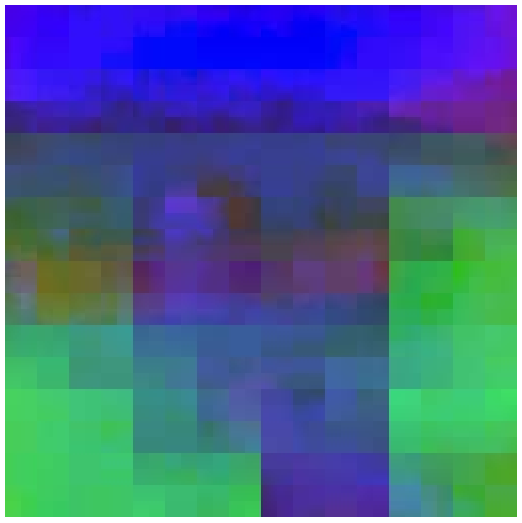
Size of the range blocks: H = 16, S = 64, and V = 8	
9) Decompressed RGB Image. 	10) Decompressed HSV Image. 
Size of the range blocks: H = 16, S = 64, and V = 16	
11) Decompressed RGB Image. 	12) Decompressed HSV Image. 

Figure A.11: (Continued)

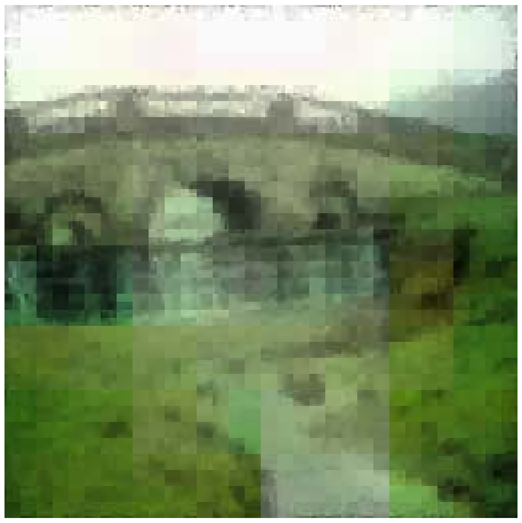
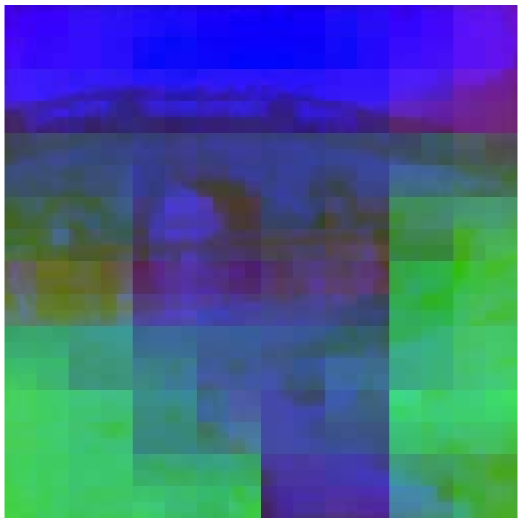
Size of the range blocks: H = 32, S = 64, and V = 8	
13) Decompressed RGB Image. 	14) Decompressed HSV Image. 

Figure A.11: (Continued)

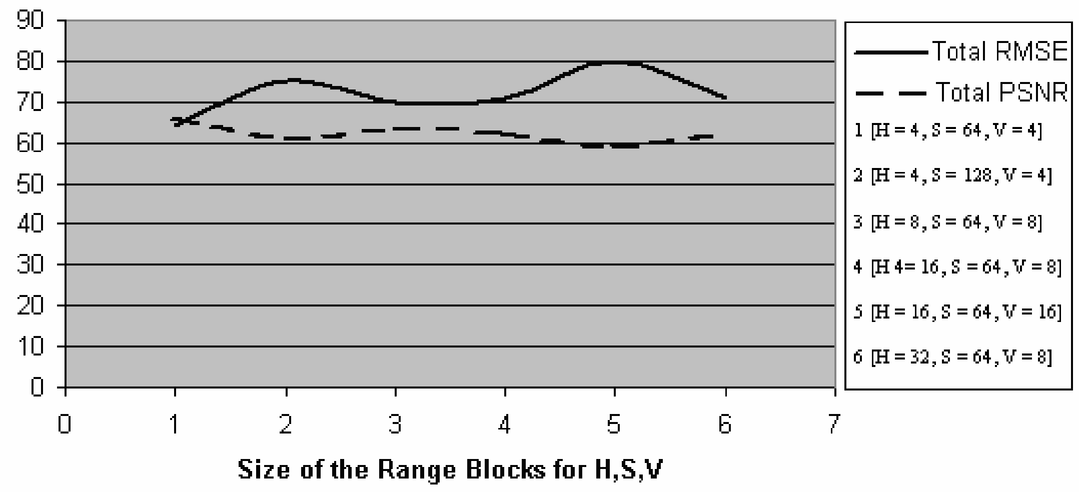


Figure A.12: RMSE and PSNR values for Image "f"

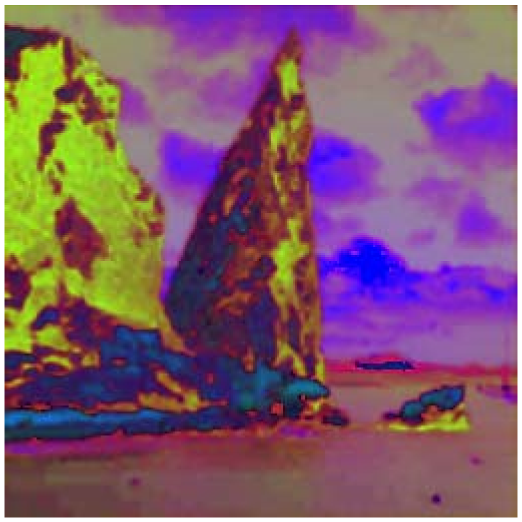
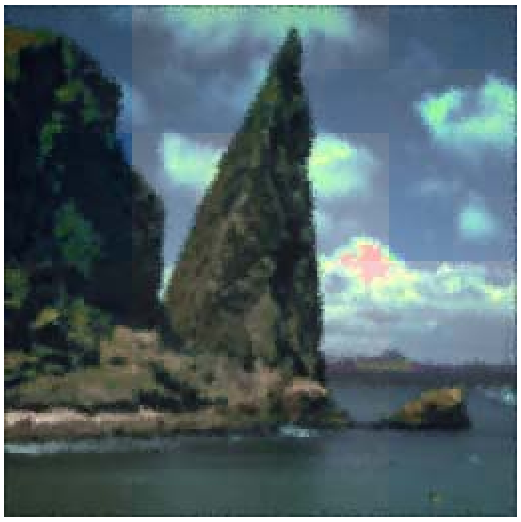
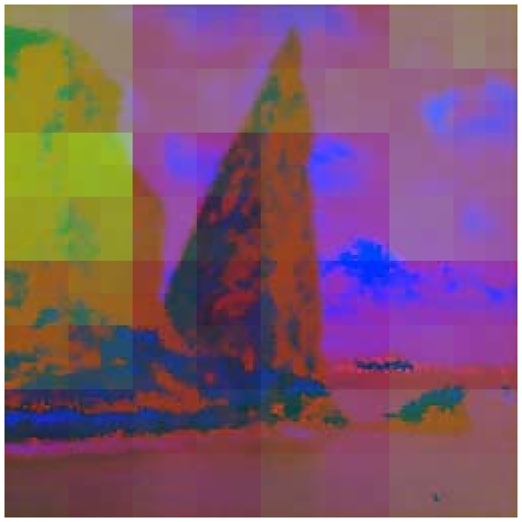
Image "g"	
1) Original RGB Image.	2) Original HSV Image. 
Size of the range blocks: H = 4, S = 64, and V = 4	
3) Decompressed RGB Image. 	4) Decompressed HSV Image. 

Figure A.13: Applying Three-Channel-HSV-Model to Image "g"

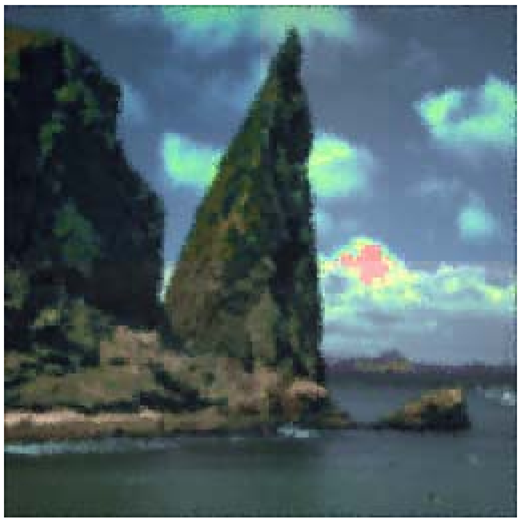
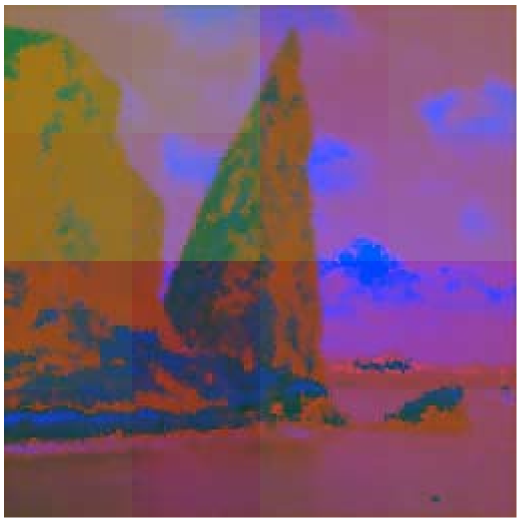
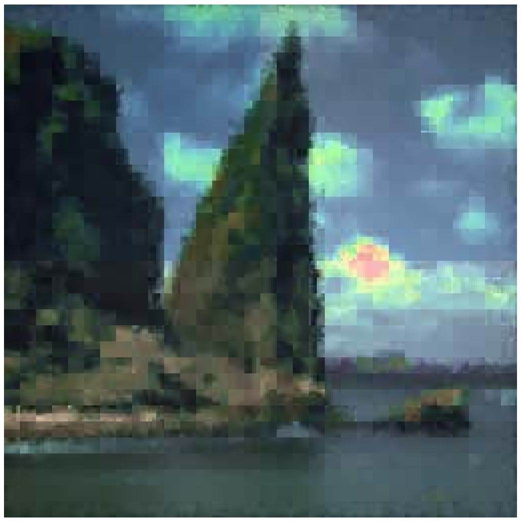
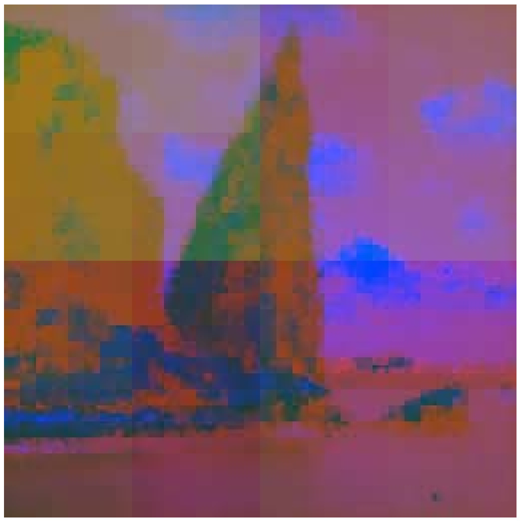
Size of the range blocks: H = 4, S = 128, and V = 4	
5) Decompressed RGB Image. 	6) Decompressed HSV Image. 
Size of the range blocks: H = 8, S = 128, and V = 8	
7) Decompressed RGB Image. 	8) Decompressed HSV Image. 

Figure A.13: (Continued)

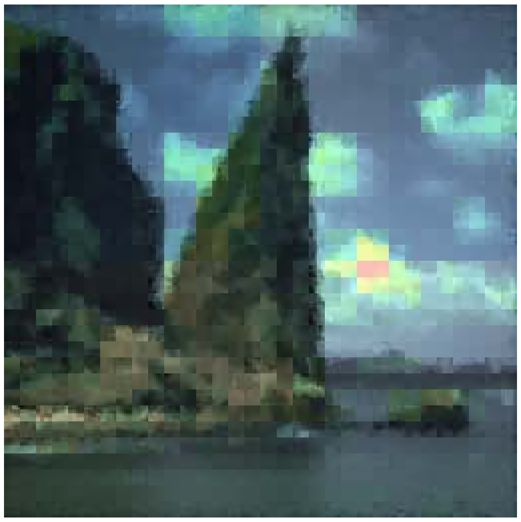
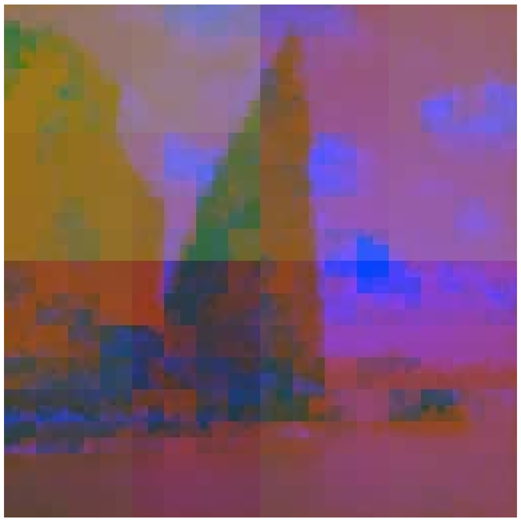
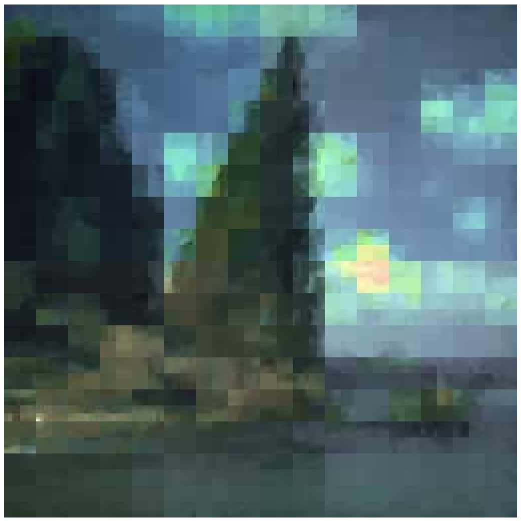
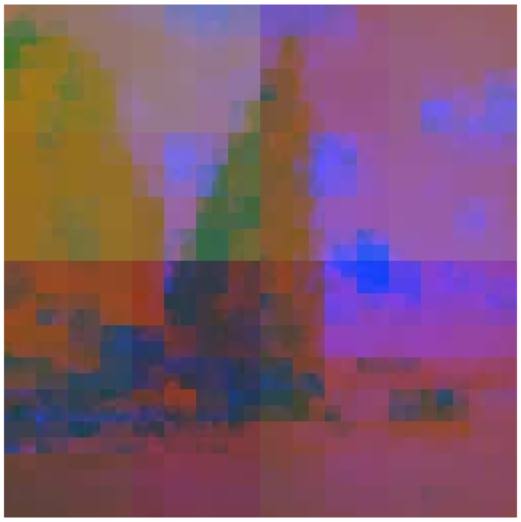
Size of the range blocks: H = 16, S = 128, and V = 8	
9) Decompressed RGB Image. 	10) Decompressed HSV Image. 
Size of the range blocks: H = 16, S = 128, and V = 16	
11) Decompressed RGB Image. 	12) Decompressed HSV Image. 

Figure A.13: (Continued)

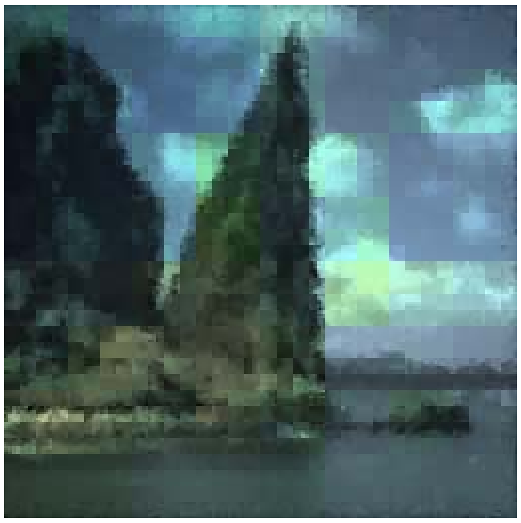
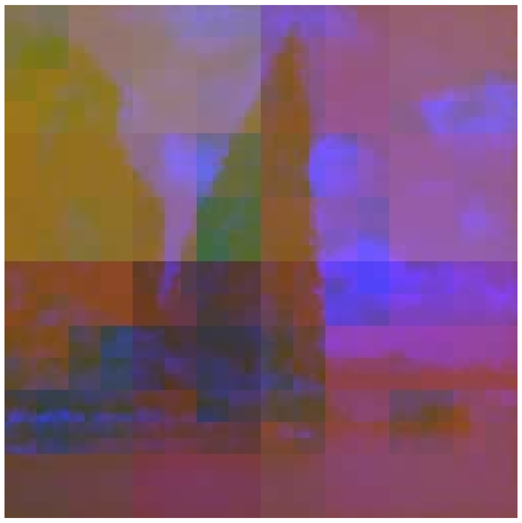
Size of the range blocks: H = 32, S = 128, and V = 8	
13) Decompressed RGB Image. 	14) Decompressed HSV Image. 

Figure A.13: (Continued)

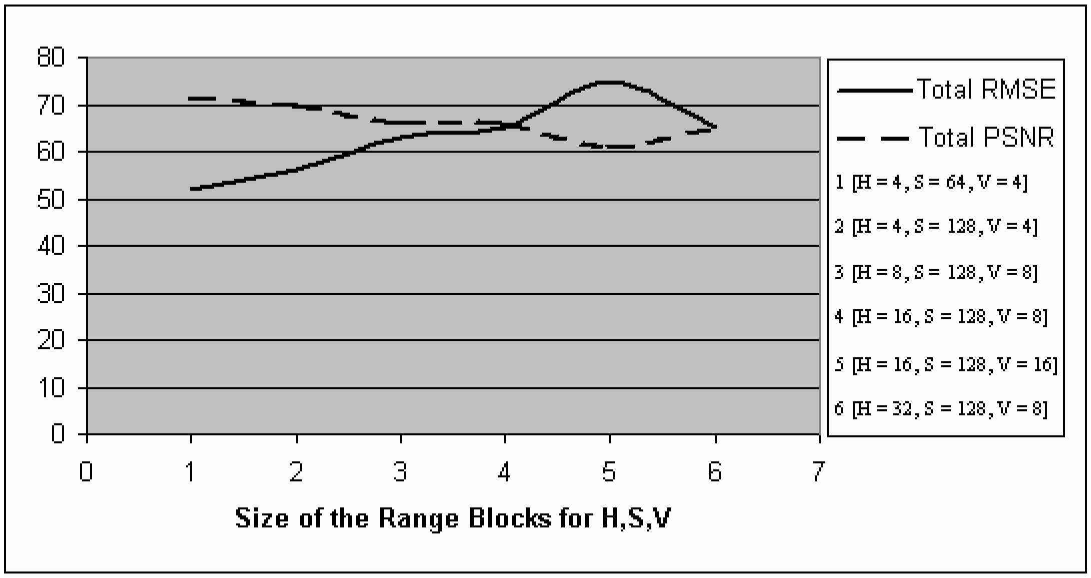


Figure A.14: RMSE and PSNR values for Image "g"

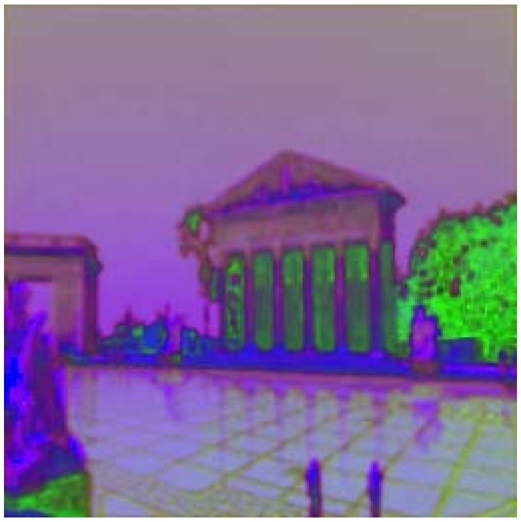
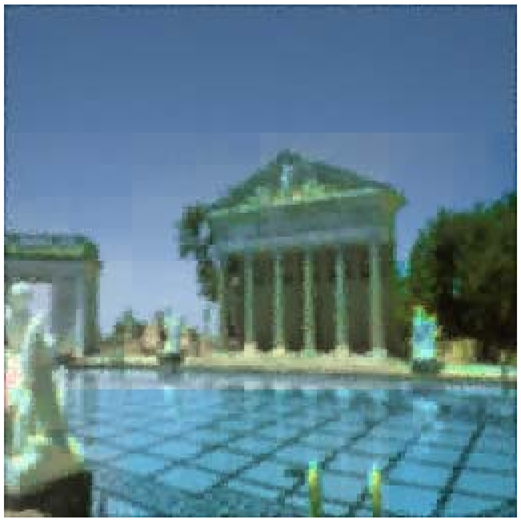
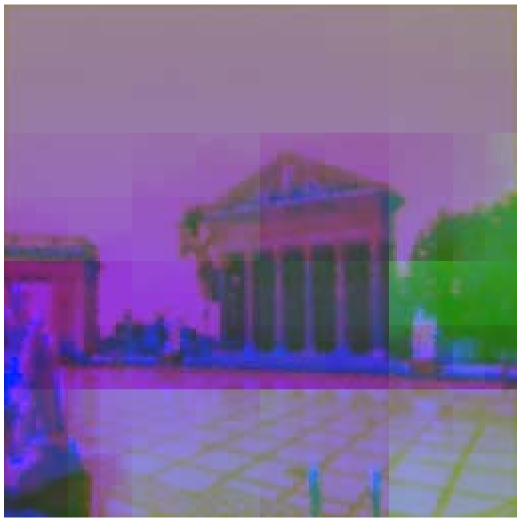
Image "h"	
1) Original RGB Image.	2) Original HSV Image. 
Size of the range blocks: H = 4, S = 64, and V = 4	
3) Decompressed RGB Image. 	4) Decompressed HSV Image. 

Figure A.15: Applying Three-Channel-HSV-Model to Image "h"

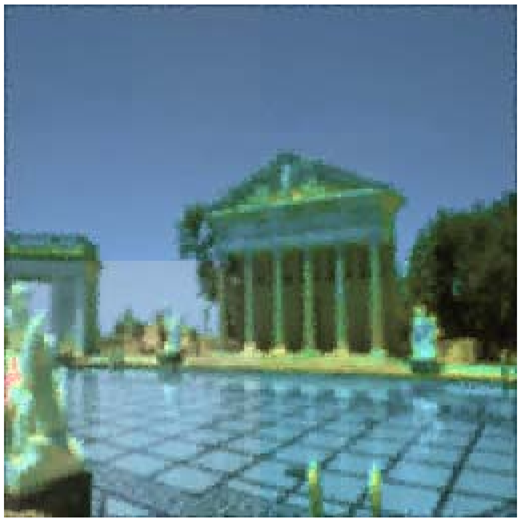
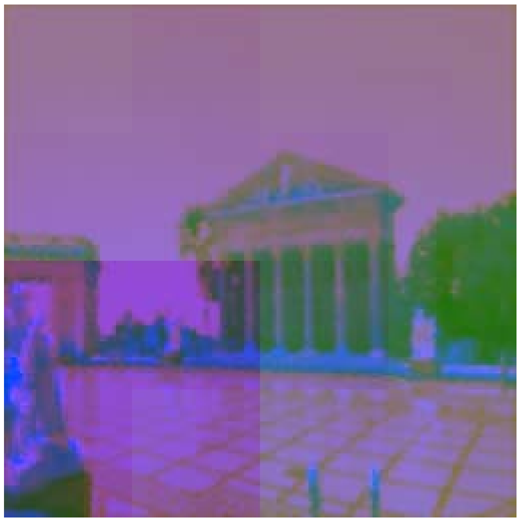
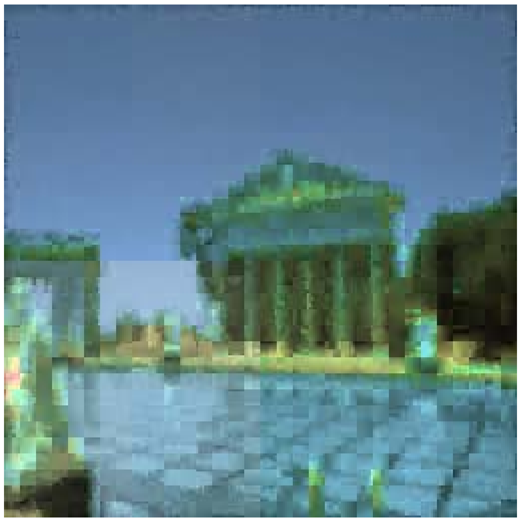
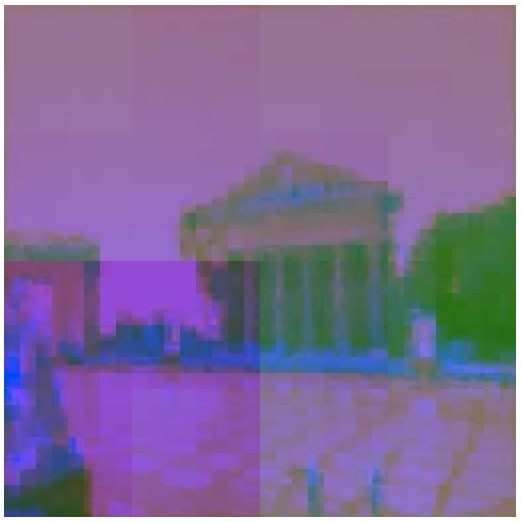
Size of the range blocks: H = 4, S = 128, and V = 4	
5) Decompressed RGB Image. 	6) Decompressed HSV Image. 
Size of the range blocks: H = 8, S = 128, and V = 8	
7) Decompressed RGB Image. 	9) Decompressed HSV Image. 

Figure A.15: (Continued)

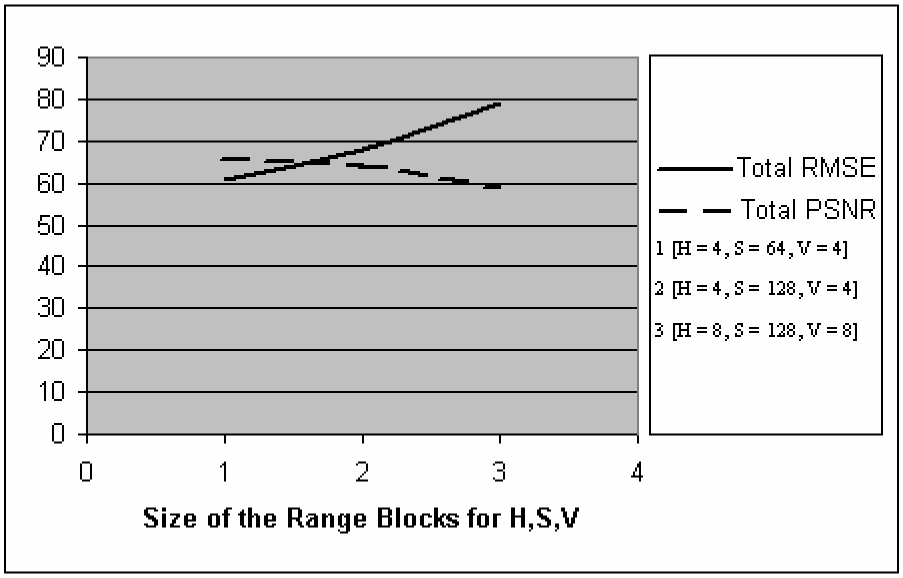


Figure A.16: RMSE and PSNR values for Image "h"

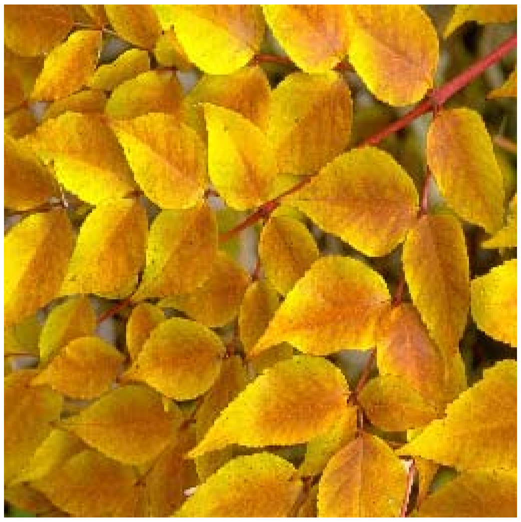
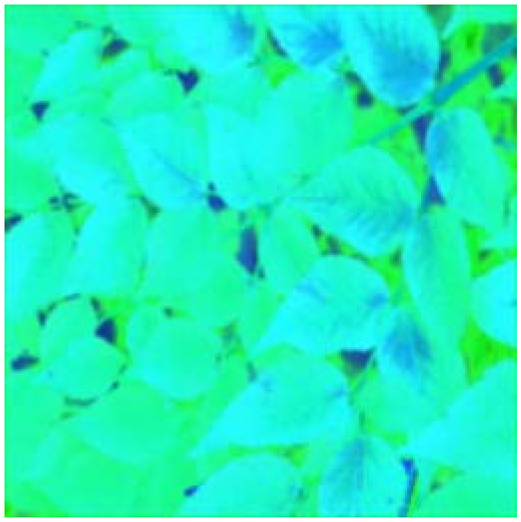
Image "i"	
1) Original RGB Image. 	2) Original HSV Image. 

Figure A.17: Applying Three-Channel-HSV-Model to Image "i"

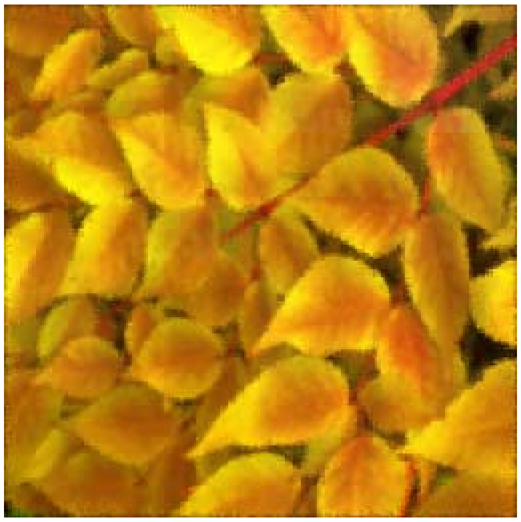
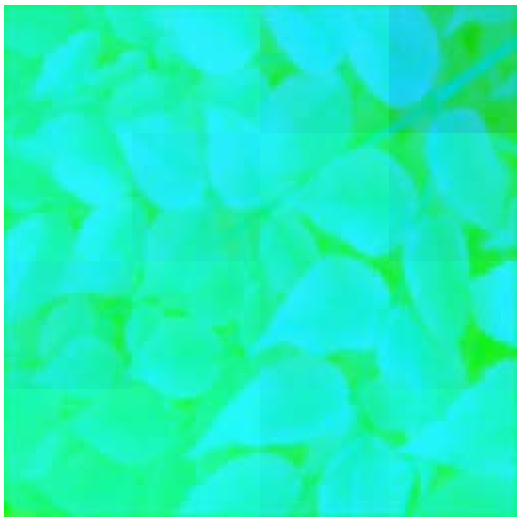
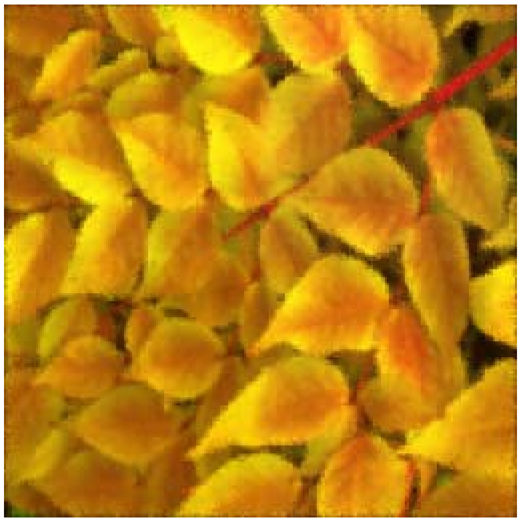
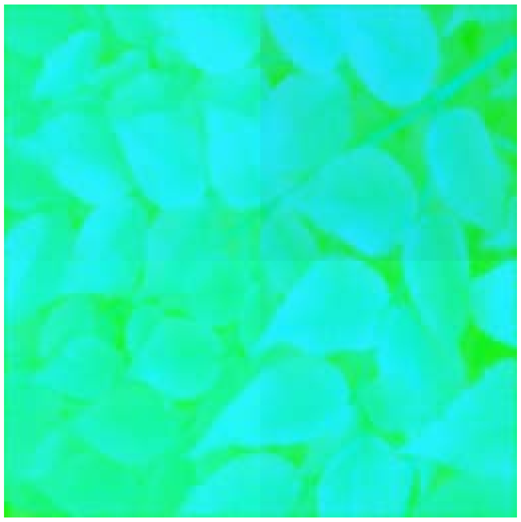
Size of the range blocks: H = 4, S = 64, and V = 4	
3) Decompressed RGB Image. 	4) Decompressed HSV Image. 
Size of the range blocks: H = 4, S = 128, and V = 4	
5) Decompressed RGB Image. 	6) Decompressed HSV Image. 

Figure A.17: (Continued)

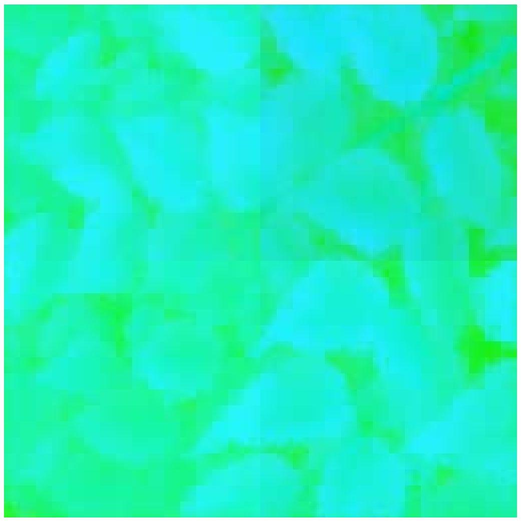
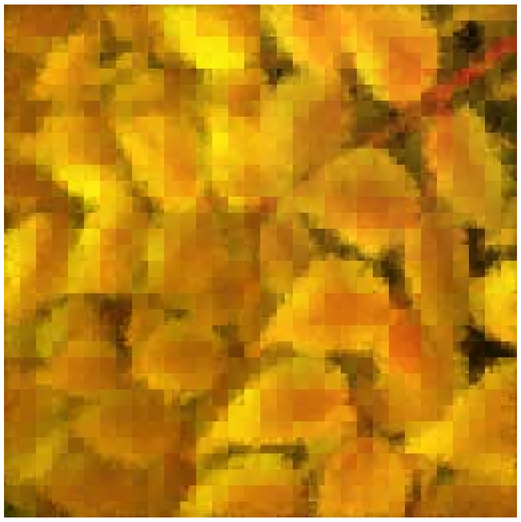
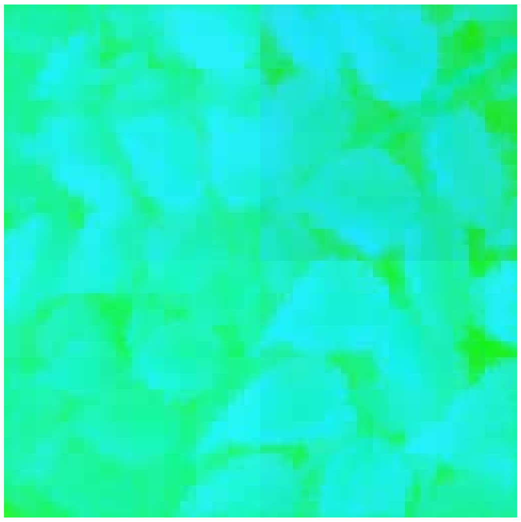
Size of the range blocks: H = 8, S = 128, and V = 8	
7) Decompressed RGB Image.	8) Decompressed HSV Image. 
Size of the range blocks: H = 16, S = 128, and V = 8	
9) Decompressed RGB Image. 	10) Decompressed HSV Image.= 

Figure A.17: (Continued)

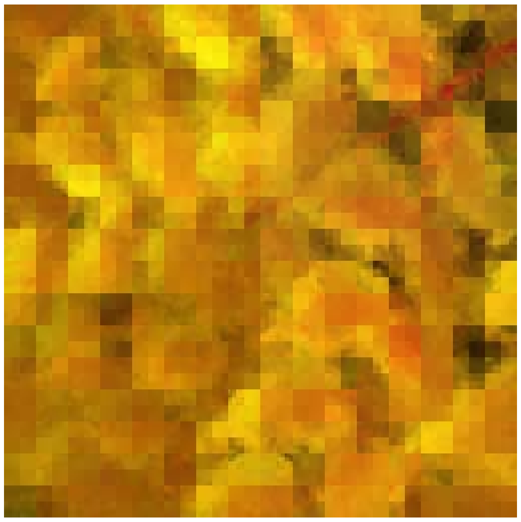
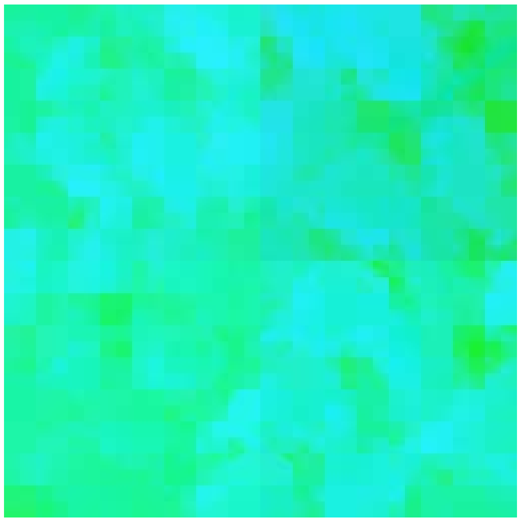
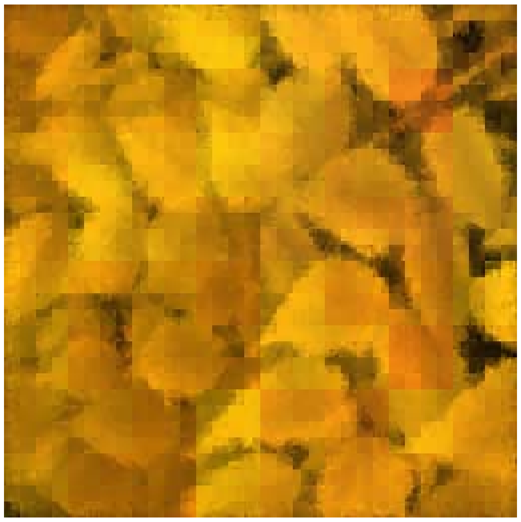
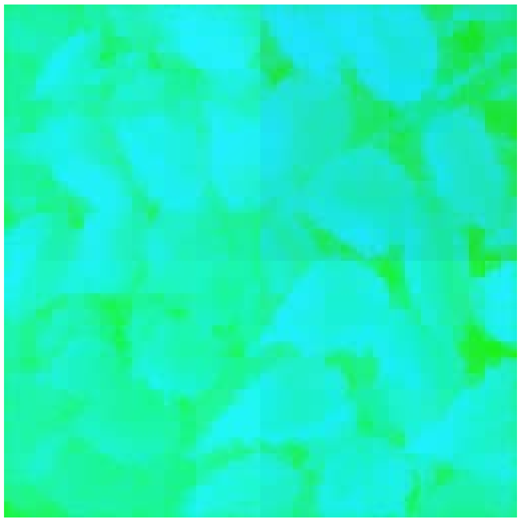
Size of the range blocks: H = 16, S = 128, and V = 16	
11) Decompressed RGB Image. 	12) Decompressed HSV Image. 
Size of the range blocks: H = 32, S =128, and V = 8	
13) Decompressed RGB Image. 	14) Decompressed HSV Image. 

Figure A.17: (Continued)

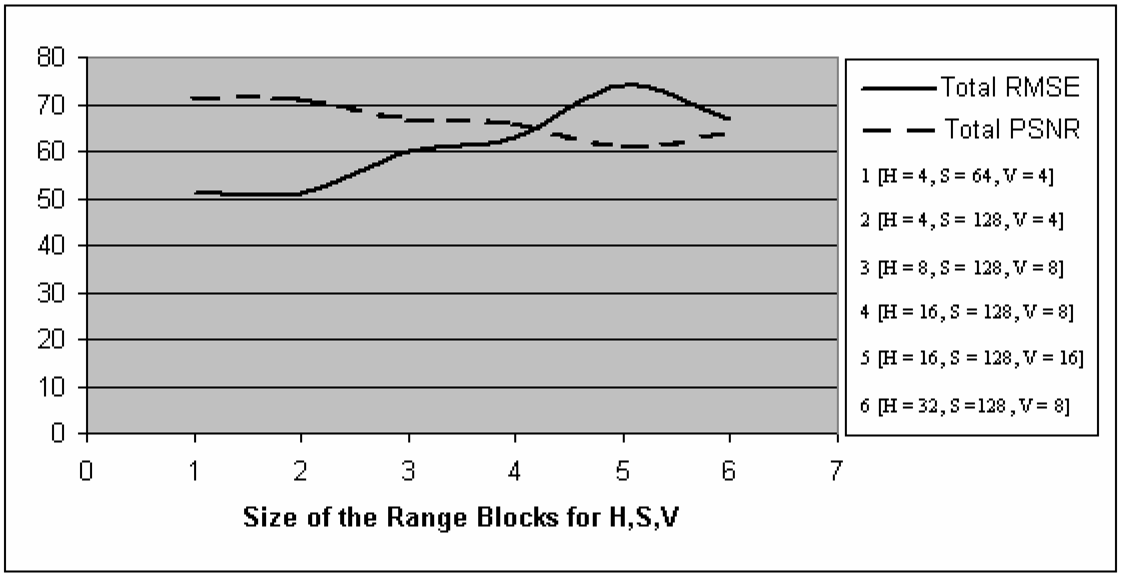


Figure A.18: RMSE and PSNR values for Image "i"

Table A.1: The results of finding the Correlation, RMSE, and PSNR between original images (a-i) and their reconstructed images after applying the Three-Channel-HSV-Model.

H bits	S bits	V bits	Red Correlation	Green Correlation	Blue Correlation	Red RMSE	Green RMSE	Blue RMSE	Total RMSE	Red PSNR	Green PSNR	Blue PSNR	Total PSNR
Image "a"													
4	32	4	0.91	0.92	0.92	13	13	13	39	26	26	26	78
4	64	4	0.91	0.92	0.92	13	13	14	40	26	26	25	77
4	128	4	0.9	0.92	0.9	14	13	16	43	25	26	24	75
8	128	8	0.87	0.9	0.88	16	15	17	48	24	25	4	53
8	128	16	0.84	0.87	0.86	17	16	18	51	24	24	23	71
16	128	8	0.86	0.9	0.88	16	15	17	48	24	25	24	73
Image "b"													
4	32	4	0.97	0.97	0.91	15	15	18	48	25	25	23	73
4	64	4	0.97	0.97	0.91	15	15	18	48	25	25	23	73
8	64	8	0.94	0.93	0.85	22	22	23	67	21	21	21	63
8	64	16	0.87	0.86	0.78	30	31	27	88	19	18	20	57
16	64	8	0.93	0.93	0.84	22	23	23	68	21	21	21	63
32	64	8	0.94	0.93	0.85	22	23	23	68	21	21	21	63
64	64	8	0.94	0.93	0.84	22	23	23	68	21	21	21	63
Image "c"													
4	64	4	0.99	0.99	0.93	9	9	13	31	29	29	26	84
4	128	4	0.99	0.98	0.84	9	11	20	40	29	27	22	78
8	128	8	0.99	0.98	0.84	10	11	20	41	28	27	22	77
8	128	16	0.99	0.98	0.84	13	12	20	45	26	27	22	75

Table A.1: (Continued)

H bits	S bits	V bits	Red Correlation	Green Correlation	Blue Correlation	Red RMSE	Green RMSE	Blue RMSE	Total RMSE	Red PSNR	Green PSNR	Blue PSNR	Total PSNR
Image “c” (Continued)													
16	128	8	0.99	0.98	0.84	10	11	20	41	28	27	22	77
16	128	16	0.99	0.98	0.84	13	12	20	45	26	27	22	75
32	128	8	0.99	0.98	0.84	10	11	20	41	28	27	22	77
64	128	8	0.99	0.98	0.84	10	13	20	43	28	26	22	76
Image "d"													
4	64	4	0.86	0.88	0.93	29	28	29	86	19	19	19	57
4	128	4	0.86	0.88	0.93	29	29	29	87	19	19	19	57
8	128	8	0.83	0.86	0.91	31	31	31	93	18	18	18	54
8	128	16	0.81	0.84	0.9	33	33	32	98	18	18	18	54
16	128	8	0.82	0.86	0.91	32	31	31	94	18	18	18	54
32	128	8	0.82	0.86	0.91	32	31	31	94	18	18	18	54
Image "e"													
4	64	4	0.96	0.98	0.89	15	10	18	43	25	28	23	76
4	128	4	0.95	0.97	0.87	16	11	20	47	24	27	22	73
8	128	8	0.93	0.96	0.85	18	13	21	52	23	26	22	71
16	128	8	0.92	0.96	0.85	20	14	21	55	22	25	22	69
16	128	16	0.9	0.93	0.82	23	18	23	64	21	23	21	65
32	128	8	0.9	0.95	0.83	23	15	22	60	21	25	21	67
Image "f"													
4	64	4	0.96	0.96	0.96	21	19	24	64	22	23	21	66
4	128	4	0.94	0.95	0.94	24	20	31	75	21	22	18	61
8	64	8	0.95	0.94	0.95	23	21	26	70	21	22	20	63
16	64	8	0.94	0.94	0.95	23	22	26	71	21	21	20	62
16	64	16	0.93	0.92	0.93	26	25	29	80	20	20	19	59
32	64	8	0.94	0.94	0.95	23	22	26	71	21	21	20	62

Table A.1: (Continued)

H bits	S bits	V bits	Red Correlation	Green Correlation	Blue Correlation	Red RMSE	Green RMSE	Blue RMSE	Total RMSE	Red PSNR	Green PSNR	Blue PSNR	Total PSNR
Image "g"													
4	64	4	0.92	0.97	0.97	21	15	16	52	22	25	24	71
4	128	4	0.91	0.96	0.96	24	15	17	56	21	25	24	70
8	128	8	0.89	0.95	0.95	26	18	19	63	20	23	23	66
16	128	8	0.88	0.95	0.95	27	19	19	65	20	23	23	66
16	128	16	0.85	0.92	0.93	30	23	22	75	19	21	21	61
32	128	8	0.88	0.94	0.96	28	19	18	65	19	23	23	65
Image "h"													
4	64	4	0.86	0.91	0.92	21	18	22	61	22	23	21	66
4	128	4	0.78	0.91	0.9	26	18	24	68	20	23	21	64
8	128	8	0.73	0.85	0.86	28	22	29	79	19	21	19	59
Image "i"													
4	64	4	0.94	0.93	0.29	16	17	18	51	24	24	23	71
4	128	4	0.94	0.93	0.21	16	17	18	51	24	24	23	71
8	128	8	0.89	0.89	0.21	21	21	18	60	22	22	23	67
16	128	8	0.89	0.85	0.21	21	24	18	63	22	21	23	66
16	128	16	0.79	0.78	0.22	28	28	18	74	19	19	23	61
32	128	8	0.89	0.8	0.21	21	28	18	67	22	19	23	64

APPENDIX –B

Results of Applying the Three-Channel-HSV-Model (Part II)

Here we show the results of applying the Three-Channel-HSV-Model over other set of (256×256, 8-bit) testing images.

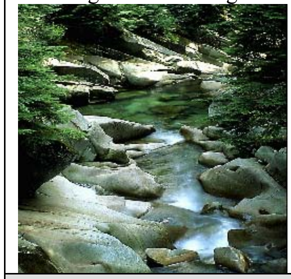
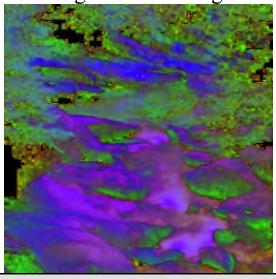
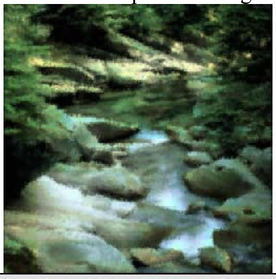
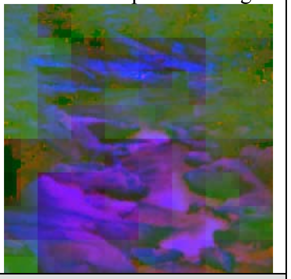
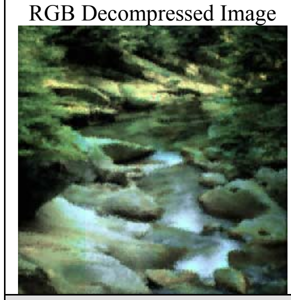
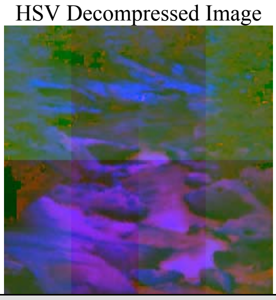
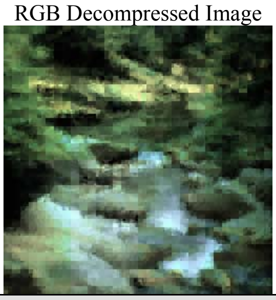
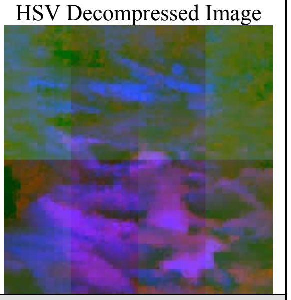
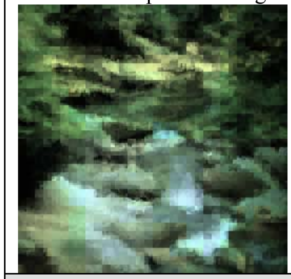
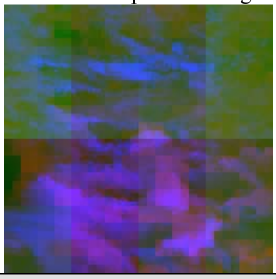
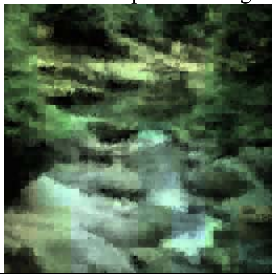
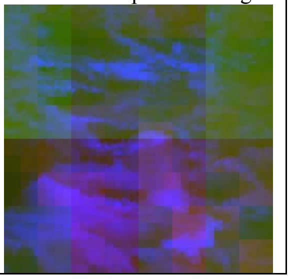
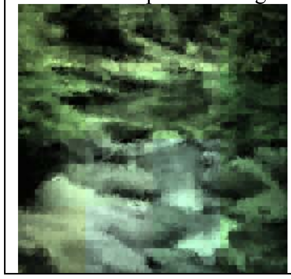
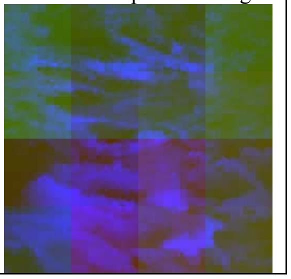
Image “1”			
Original Images		Size of the range blocks: H = 4 , S = 32 , and V =4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4 , S = 64 , and V =4		Size of the range blocks: H = 8 , S = 64 , and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 16 , S = 64 , and V = 8		Size of the range blocks: H = 32 , S = 64 , and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 64 , S = 64 , and V = 8			
RGB Decompressed Image	HSV Decompressed Image		
			

Figure B.1: Applying Three-Channel-HSV-Model to image "1"

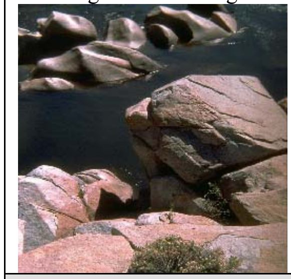
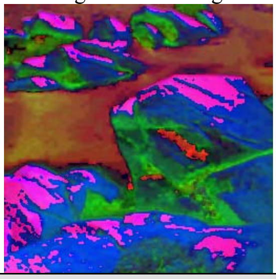
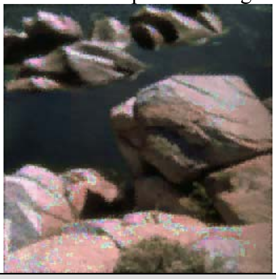
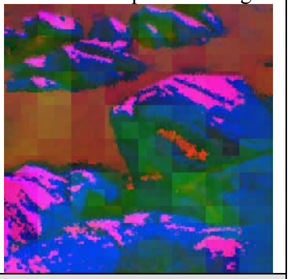
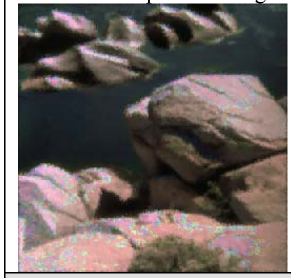
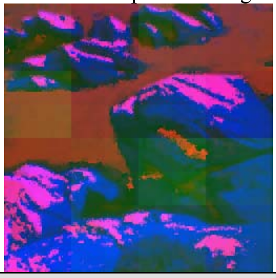
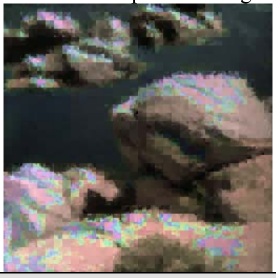
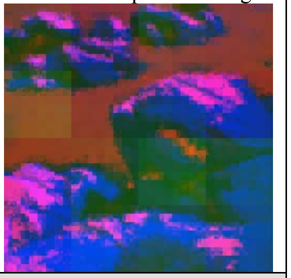
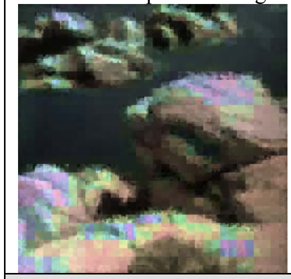
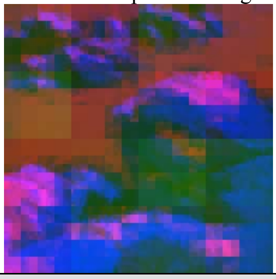
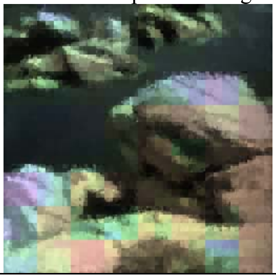
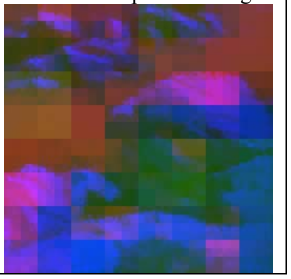
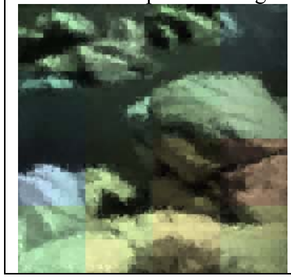
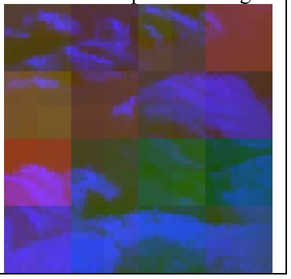
Image “2 ”			
Original Images		Size of the range blocks: H = 4 , S = 32 , and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4 , S = 64 , and V = 4		Size of the range blocks: H = 8 , S = 64 , and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 16 , S = 64 , and V = 8		Size of the range blocks: H = 32 , S = 64 , and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 64 , S = 64 , and V = 8			
RGB Decompressed Image	HSV Decompressed Image		
			

Figure B.2: Applying Three-Channel-HSV-Model to image "2"

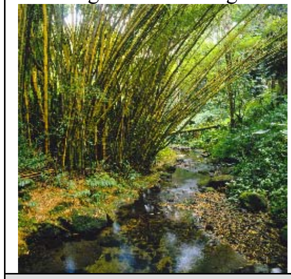
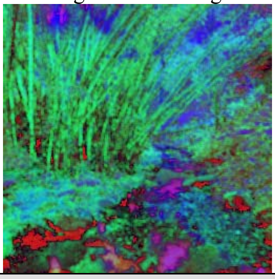
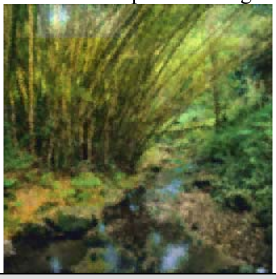
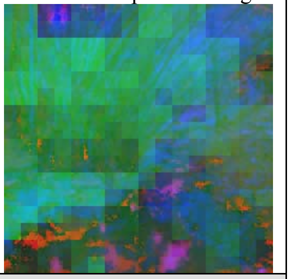
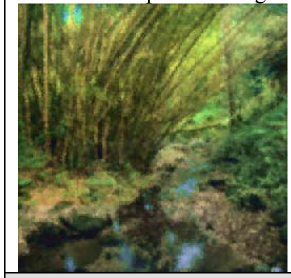
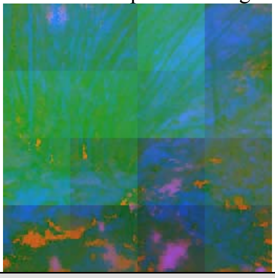
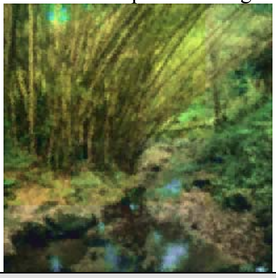
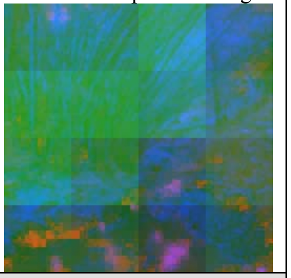
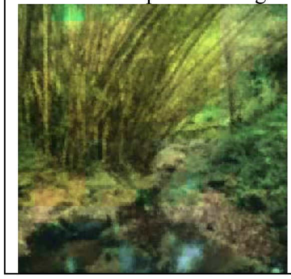
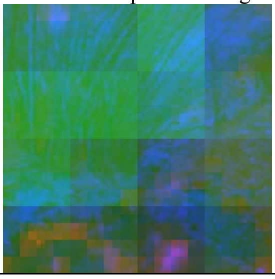
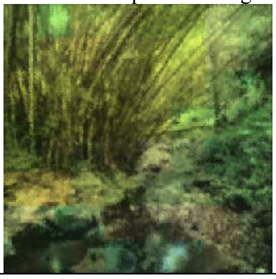
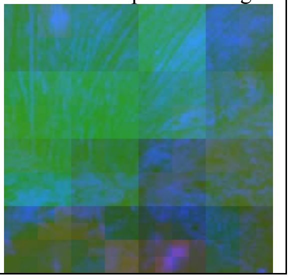
Image “3”			
Original Images		Size of the range blocks: H = 4 , S = 32 , and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4 , S = 64 , and V = 4		Size of the range blocks: H = 8 , S = 64 , and V = 4	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 16 , S = 64 , and V = 4		Size of the range blocks: H = 32 , S = 64 , and V = 4	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			

Figure B.3: Applying Three-Channel-HSV-Model to image "3"

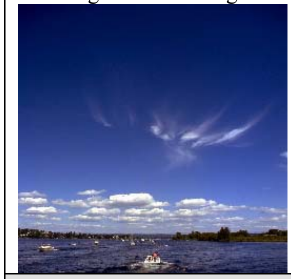
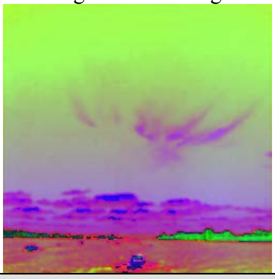
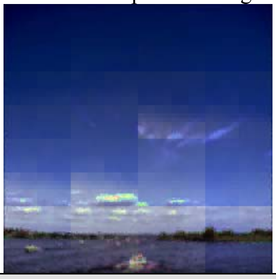
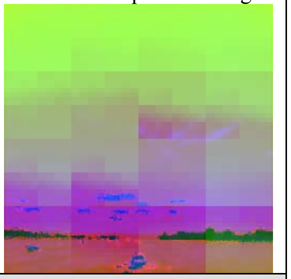
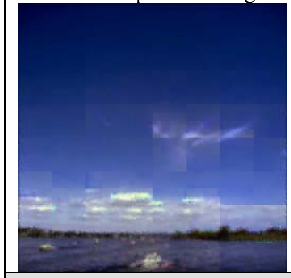
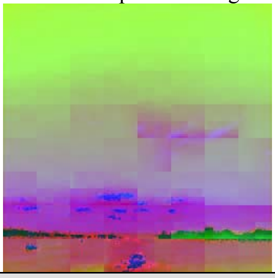
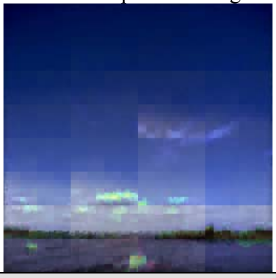
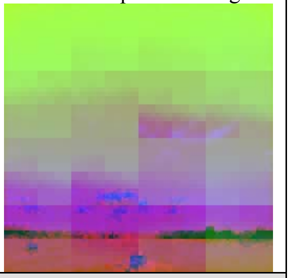
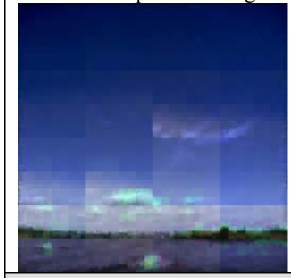
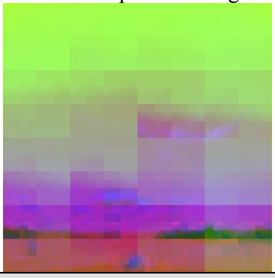
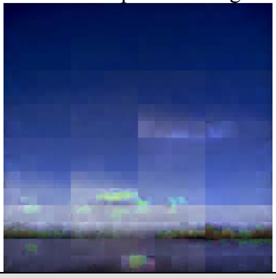
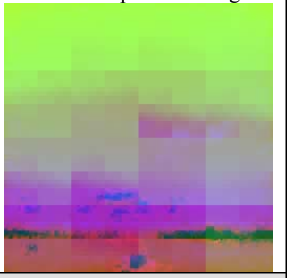
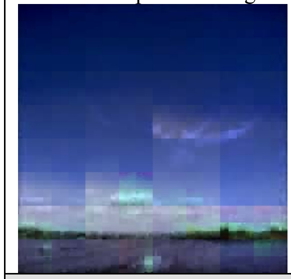
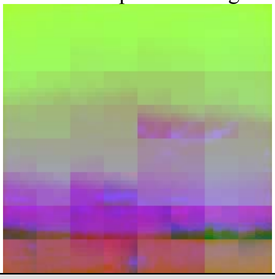
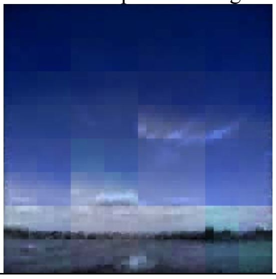
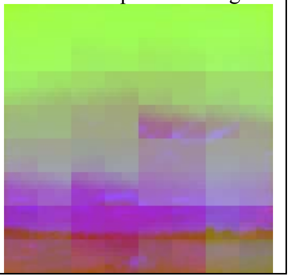
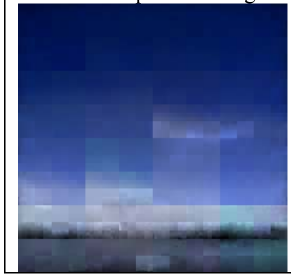
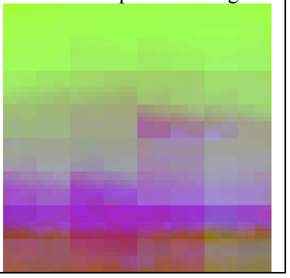
Image “4”			
Original Images		Size of the range blocks: H = 4 , S = 64 , and V = 4	
Original RGB Image 	Original HSV Image 	RGB Decompressed Image 	HSV Decompressed Image 
Size of the range blocks: H = 4 , S = 32 , and V = 4		Size of the range blocks: H = 8 , S = 64 , and V = 8	
RGB Decompressed Image 	HSV Decompressed Image 	RGB Decompressed Image 	HSV Decompressed Image 
Size of the range blocks: H = 16 , S = 64 , and V = 8		Size of the range blocks: H = 8 , S = 64 , and V = 16	
RGB Decompressed Image 	HSV Decompressed Image 	RGB Decompressed Image 	HSV Decompressed Image 
Size of the range blocks: H = 32 , S = 64 , and V = 8		Size of the range blocks: H = 64 , S = 64 , and V = 8	
RGB Decompressed Image 	HSV Decompressed Image 	RGB Decompressed Image 	HSV Decompressed Image 
Size of the range blocks: H = 64 , S = 64 , and V = 16			
RGB Decompressed Image 	HSV Decompressed Image 		

Figure B.4: Applying Three-Channel-HSV-Model to image "4"

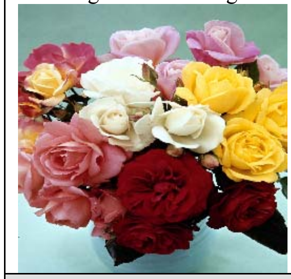
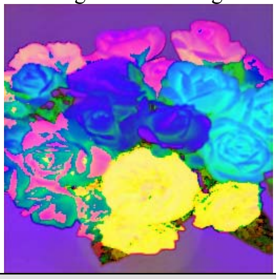
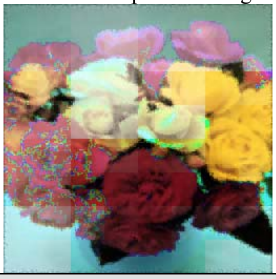
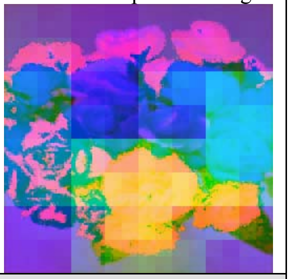
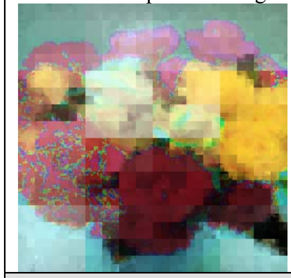
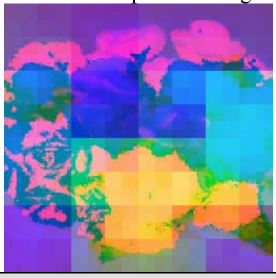
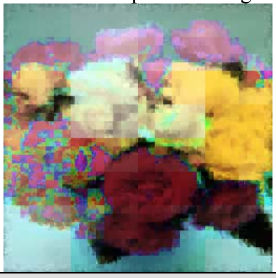
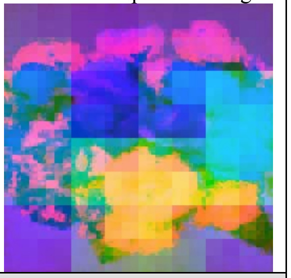
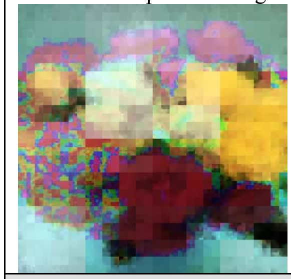
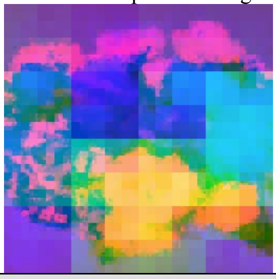
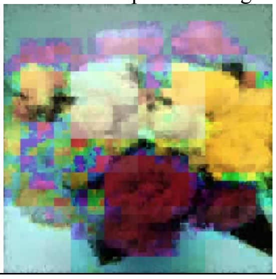
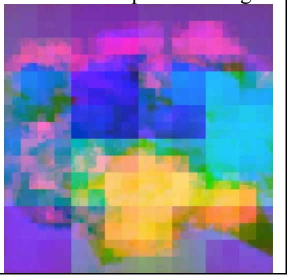
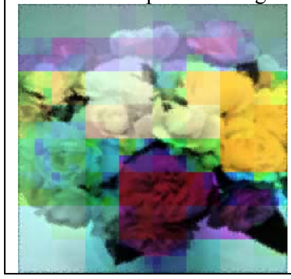
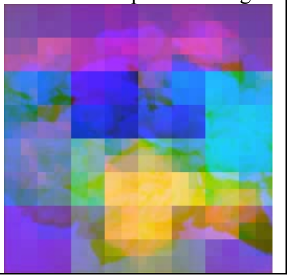
Image “5”			
Original Images		Size of the range blocks: H = 4 , S = 64 , and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4 , S = 64 , and V = 16		Size of the range blocks: H = 8 , S = 64 , and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 8 , S = 64 , and V = 16		Size of the range blocks: H = 16 , S = 64 , and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 32 , S = 64 , and V = 4			
RGB Decompressed Image	HSV Decompressed Image		
			

Figure B.5: Applying Three-Channel-HSV-Model to image "5"

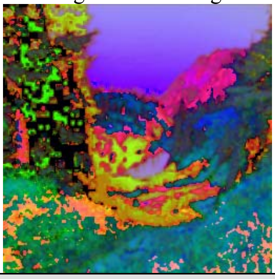
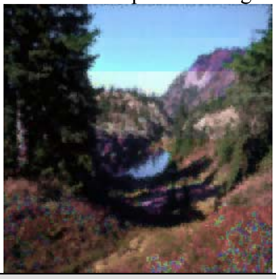
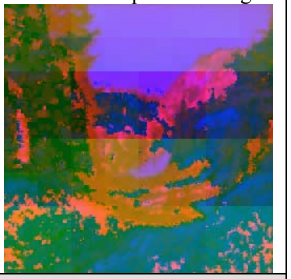
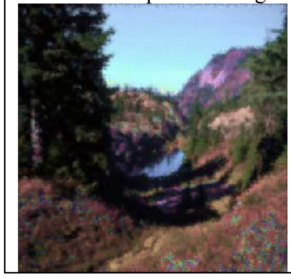
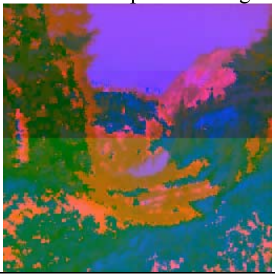
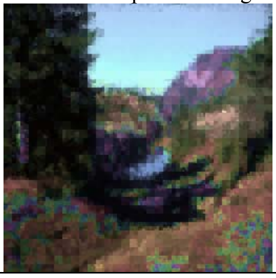
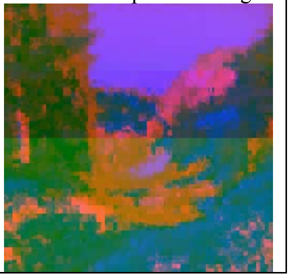
Image “6”			
Original Images		Size of the range blocks: H = 4 , S = 64 , and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4 , S = 128, and V = 4		Size of the range blocks: H = 8 , S = 128, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			

Figure B.6: Applying Three-Channel-HSV-Model to image "6"

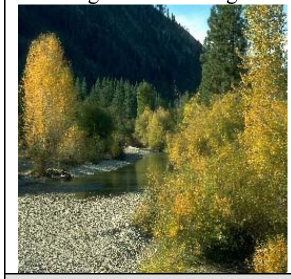
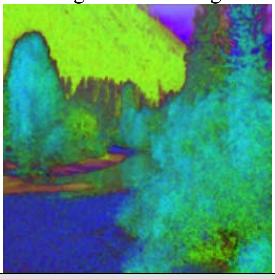
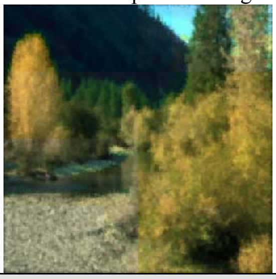
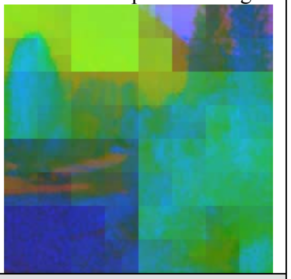
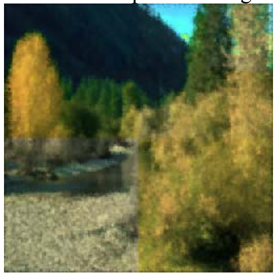
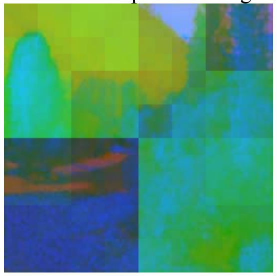
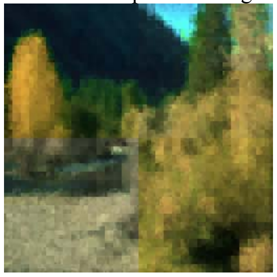
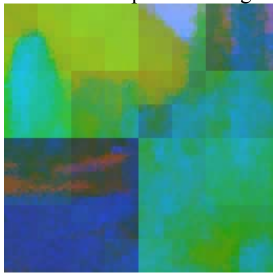
Image “7”			
Original Images		Size of the range blocks: H = 4 , S = 64 , and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4 , S = 128, and V = 4		Size of the range blocks: H = 8, S = 128, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			

Figure B.7: Applying Three-Channel-HSV-Model to image "7"

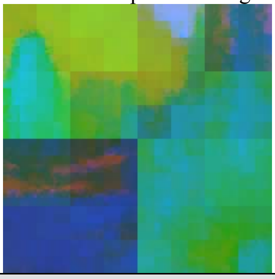
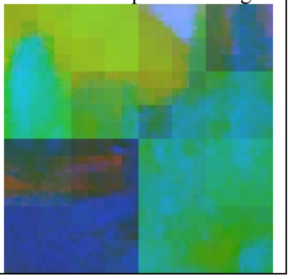
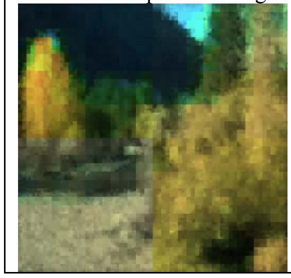
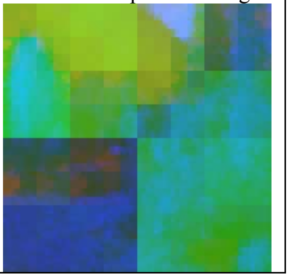
Image “7” (Continued)			
Size of the range blocks: H = 8, S = 128, and V = 16		Size of the range blocks: H = 16, S = 128, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 32, S = 128, and V = 8			
RGB Decompressed Image	HSV Decompressed Image		
			

Figure B.7: (Continued)

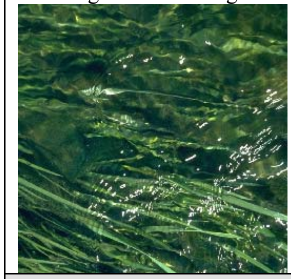
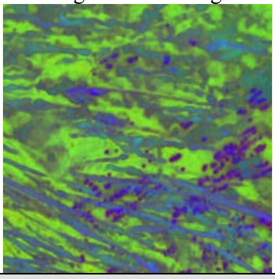
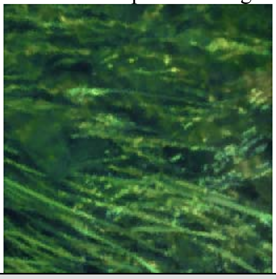
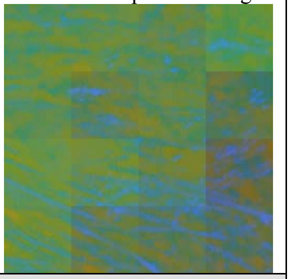
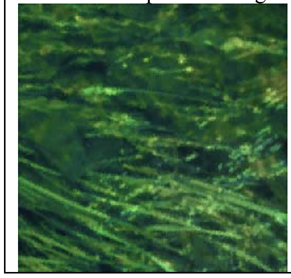
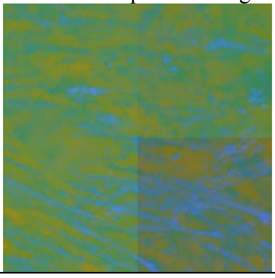
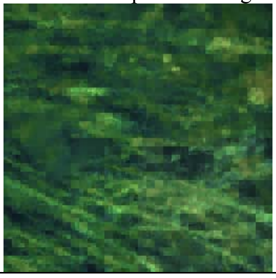
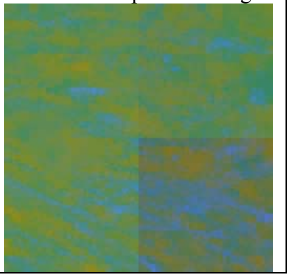
Image “8”			
Original Images		Size of the range blocks: H = 4, S = 64, and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4, S = 128, and V = 4		Size of the range blocks: H = 8, S = 128, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			

Figure B.8: Applying Three-Channel-HSV-Model to image "8"

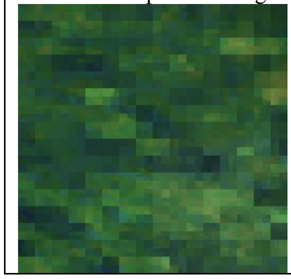
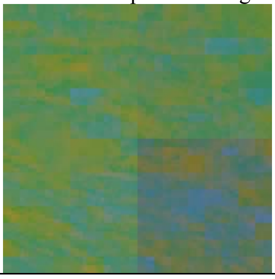
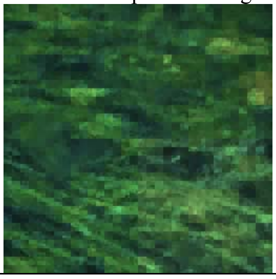
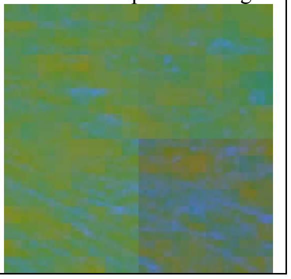
Image “8” (Continued)			
Size of the range blocks: H = 8, S = 128, and V = 16		Size of the range blocks: H = 16, S = 128, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			

Figure B.8: (Continued)

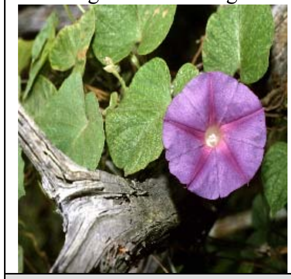
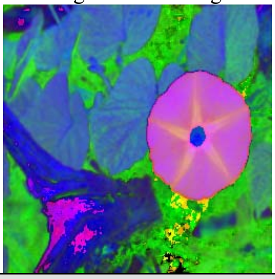
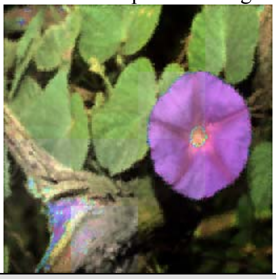
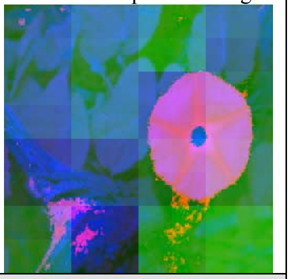
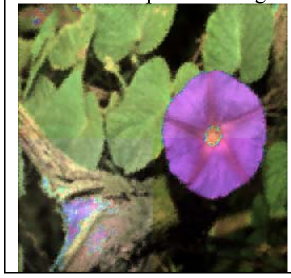
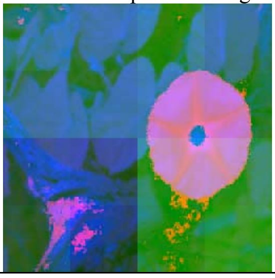
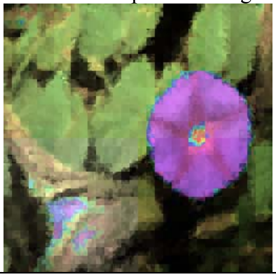
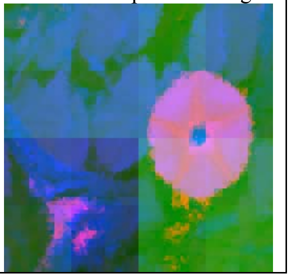
Image “9”			
Original Images		Size of the range blocks: H = 4, S = 64, and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4, S = 128, and V = 4		Size of the range blocks: H = 8, S = 128, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			

Figure B.9: Applying Three-Channel-HSV-Model to image "9"

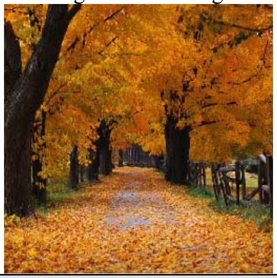
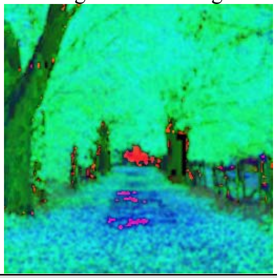
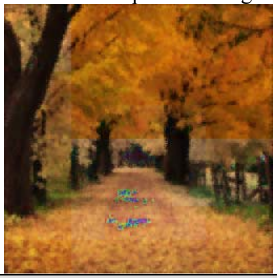
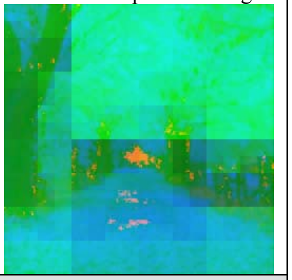
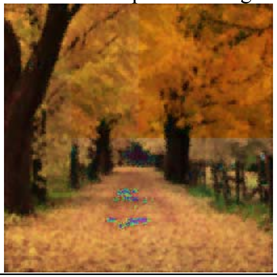
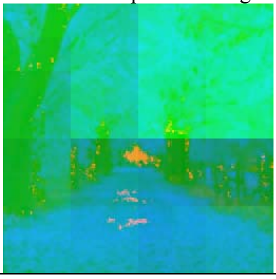
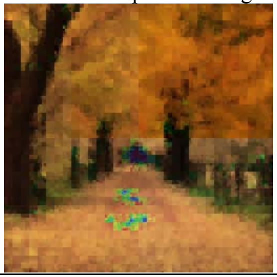
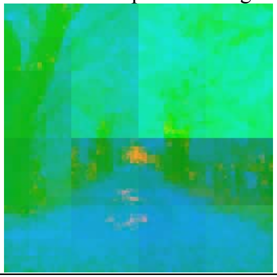
Image “10”			
Original Images		Size of the range blocks: H = 4, S = 64, and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4, S = 128, and V = 4		Size of the range blocks: H = 8, S = 128, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			

Figure B.10: Applying Three-Channel-HSV-Model to image "10"

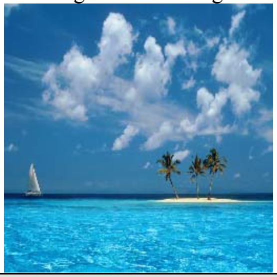
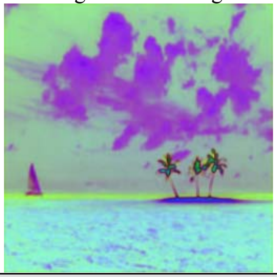
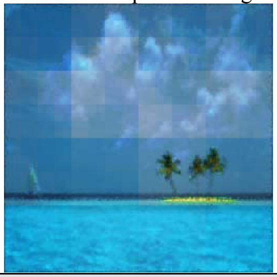
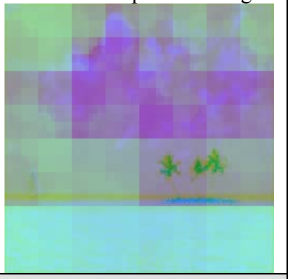
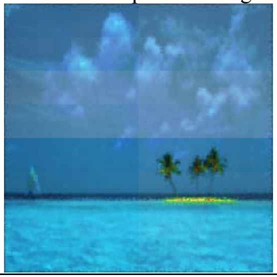
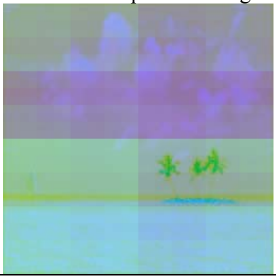
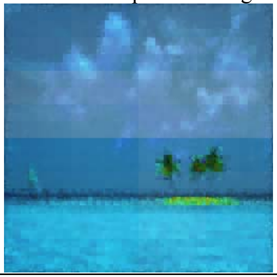
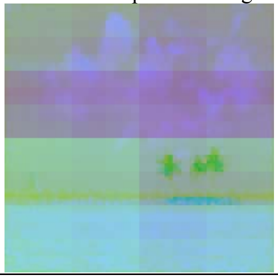
Image “11”			
Original Images		Size of the range blocks: H = 4, S = 64, and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4, S = 128, and V = 4		Size of the range blocks: H = 8, S = 128, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			

Figure B.11: Applying Three-Channel-HSV-Model to image "11"

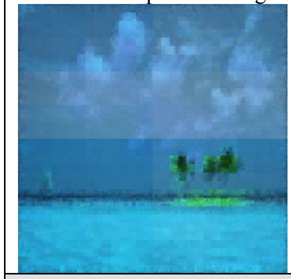
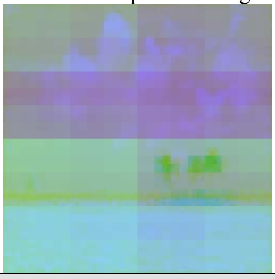
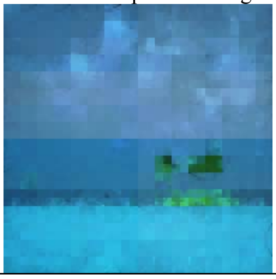
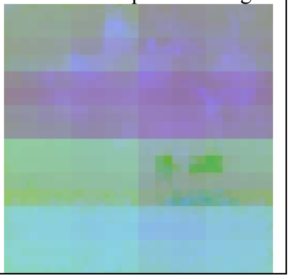
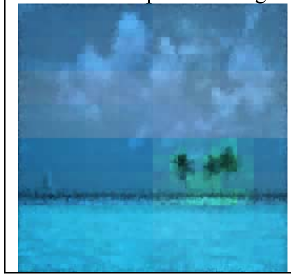
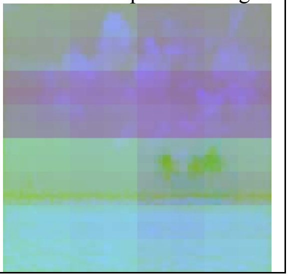
Image “11” (Continued)			
Size of the range blocks: H = 16, S = 128, and V = 8		Size of the range blocks: H = 16, S = 128, and V = 16	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 32, S = 128, and V = 8			
RGB Decompressed Image	HSV Decompressed Image		
			

Figure B.11: (Continued)

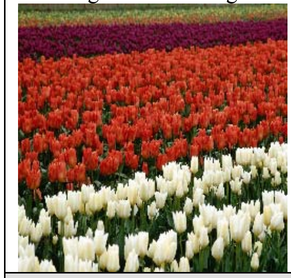
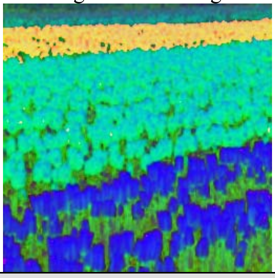
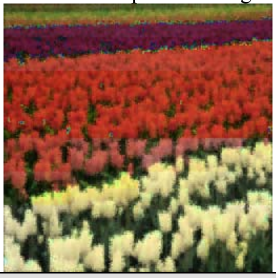
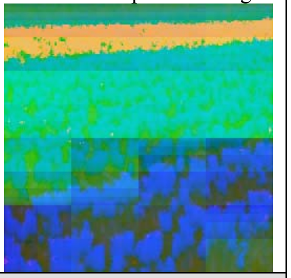
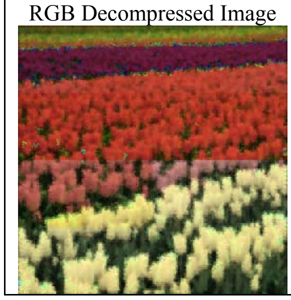
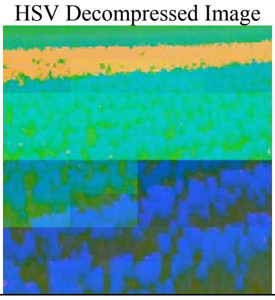
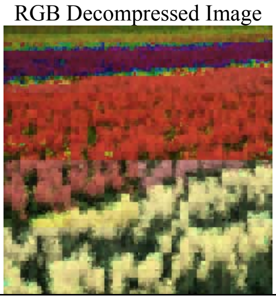
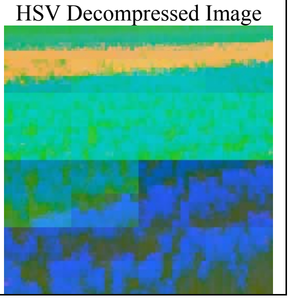
Image “12”			
Original Images		Size of the range blocks: H = 4, S = 64, and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4, S = 128, and V = 4		Size of the range blocks: H = 8, S = 128, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			

Figure B.12: Applying Three-Channel-HSV-Model to image "12"

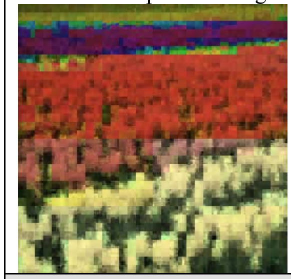
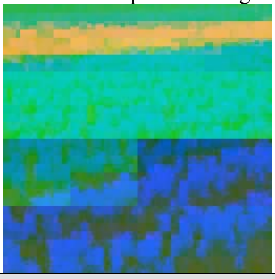
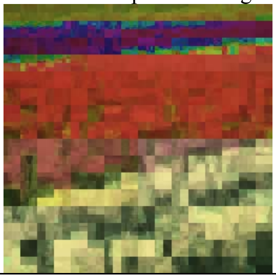
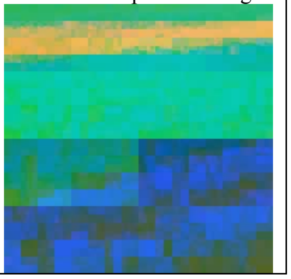
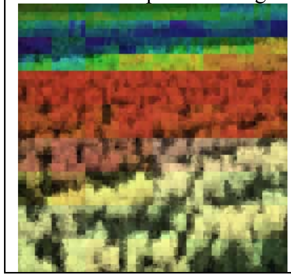
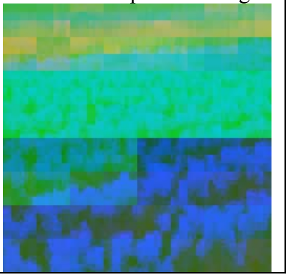
Image “12” (Continued)			
Size of the range blocks: H = 16, S= 128, V = 8		Size of the range blocks: H = 16, S= 128, and V = 16	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 32, S = 128, V= 8			
RGB Decompressed Image	HSV Decompressed Image		
			

Figure B.12: (Continued)

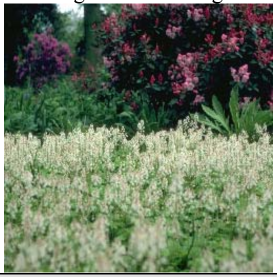
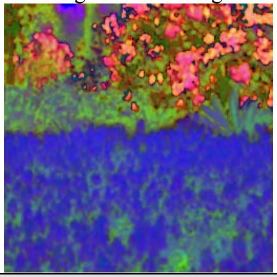
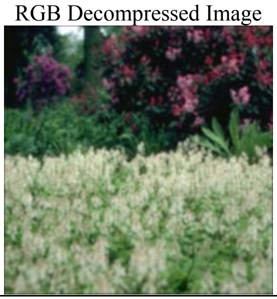
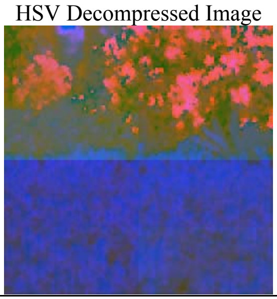
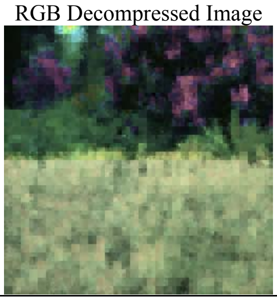
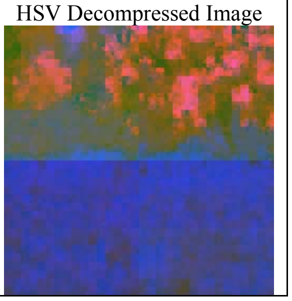
Image “13”			
Original Images		Size of the range blocks: H = 4, S = 64, and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4, S = 128, and V = 4		Size of the range blocks: H = 8, S = 128, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			

Figure B.13: Applying Three-Channel-HSV-Model to image "13"

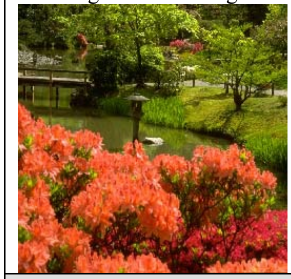
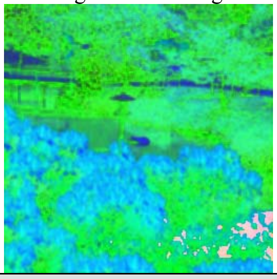
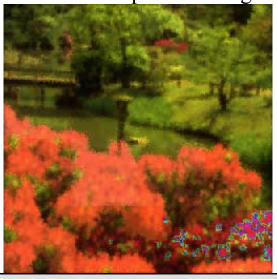
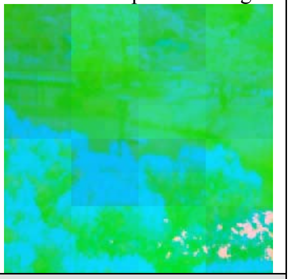
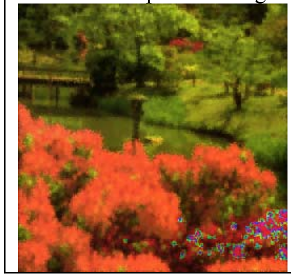
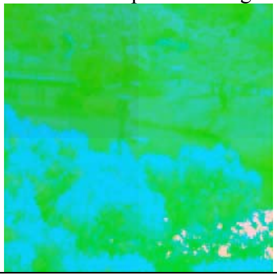
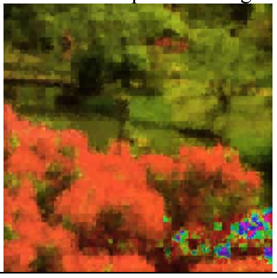
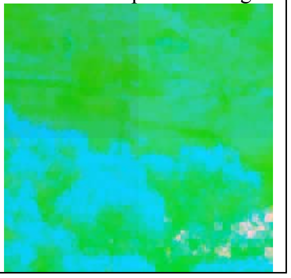
Image "14"			
Original Images		Size of the range blocks: H = 4, S = 64, and V = 4	
Original RGB Image 	Original HSV Image 	RGB Decompressed Image 	HSV Decompressed Image 
Size of the range blocks: H = 4, S = 128, and V = 4		Size of the range blocks: H = 8, S = 128, and V = 8	
RGB Decompressed Image 	HSV Decompressed Image 	RGB Decompressed Image 	HSV Decompressed Image 

Figure B.14: Applying Three-Channel-HSV-Model to image "14"

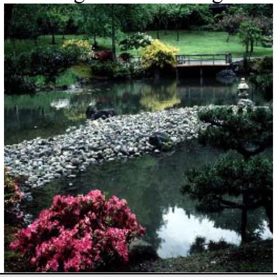
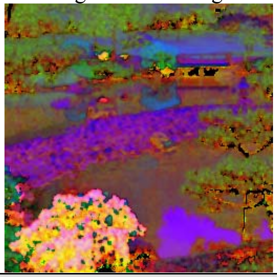
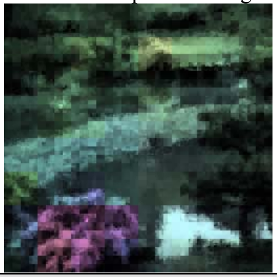
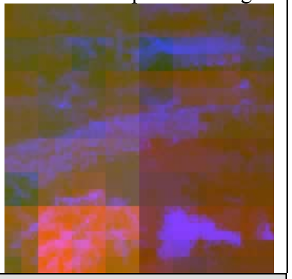
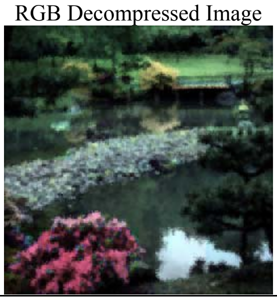
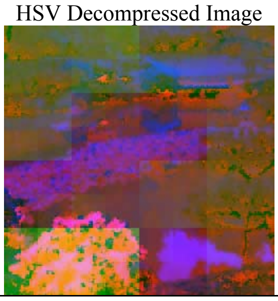
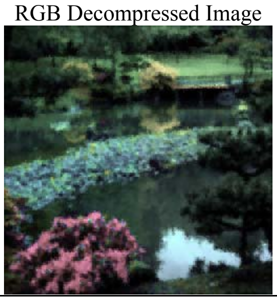
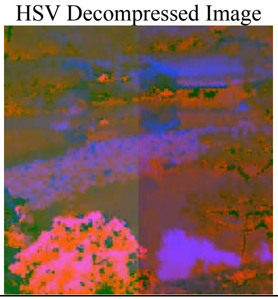
Image "15"			
Original Images		Size of the range blocks: H = 32, S = 128, and V = 8	
Original RGB Image 	Original HSV Image 	RGB Decompressed Image 	HSV Decompressed Image 
Size of the range blocks: H = 4, S = 64, and V = 4		Size of the range blocks: H = 4, S = 128, and V = 4	
RGB Decompressed Image 	HSV Decompressed Image 	RGB Decompressed Image 	HSV Decompressed Image 

Figure B.15: Applying Three-Channel-HSV-Model to image "15"

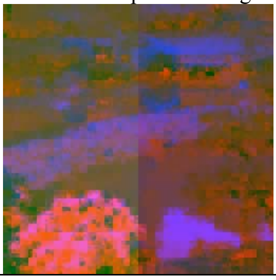
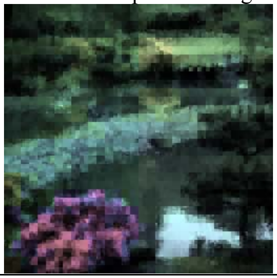
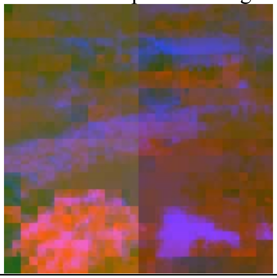
Image “15” (Continued)			
Size of the range blocks: H = 8, S = 128, and V = 8		Size of the range blocks: H = 16, S = 128, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			

Figure B.15: (Continued)

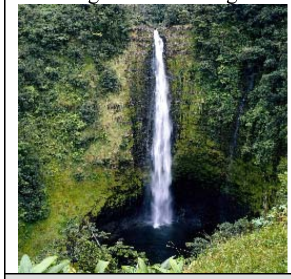
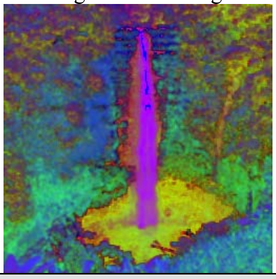
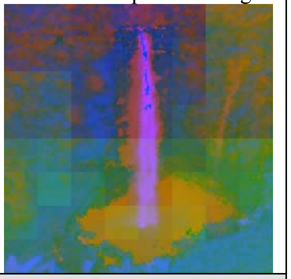
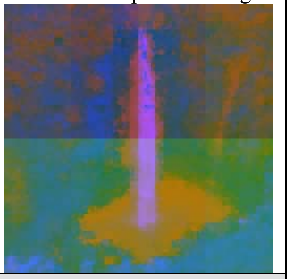
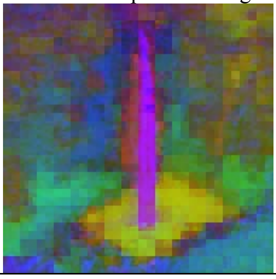
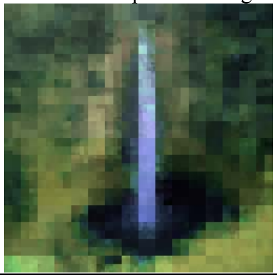
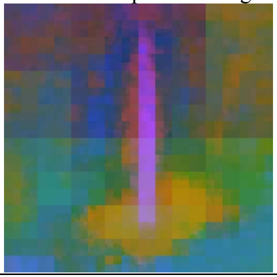
Image “16”			
Original Images		Size of the range blocks: H = 4, S = 64, and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4, S = 128, and V = 4		Size of the range blocks: H = 8, S = 128, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 16, S = 16, and V = 4		Size of the range blocks: H = 16, S = 64, and V = 16	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			

Figure B.16: Applying Three-Channel-HSV-Model to image "16"

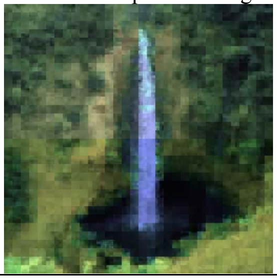
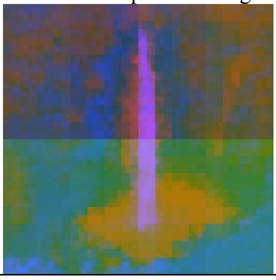
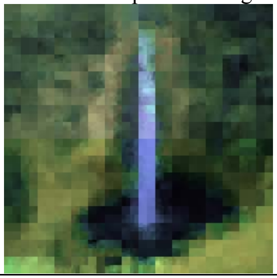
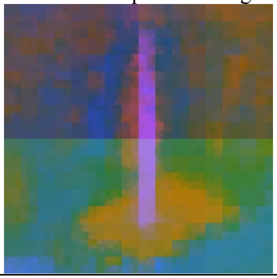
Image “16” (Continued)			
Size of the range blocks: H = 16, S = 128, and V = 8		Size of the range blocks: H = 16, S = 128, and V = 16	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			

Figure B.16: (Continued)

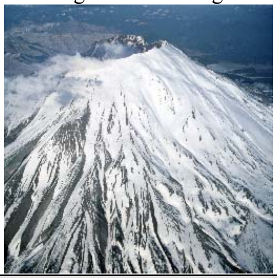
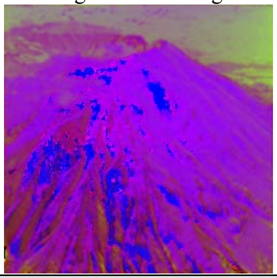
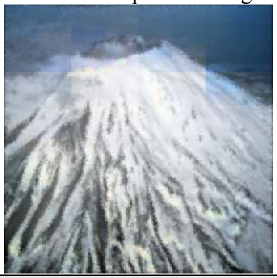
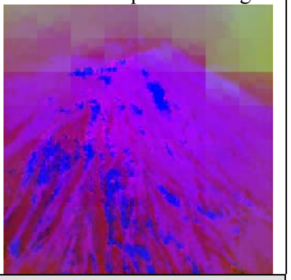
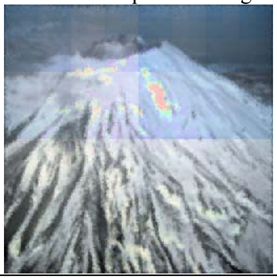
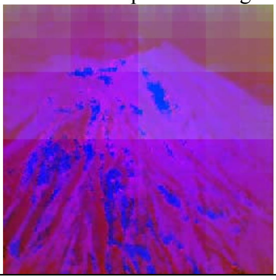
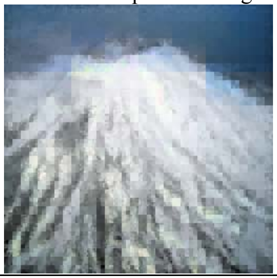
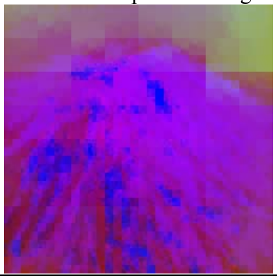
Image “17”			
Original Images		Size of the range blocks: H = 4, S = 64, and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4, S = 128, and V = 4		Size of the range blocks: H = 8, S = 64, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			

Figure B.17: Applying Three-Channel-HSV-Model to image "17"

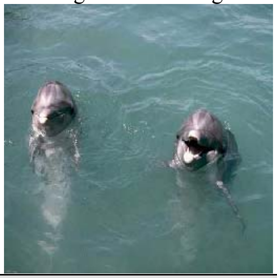
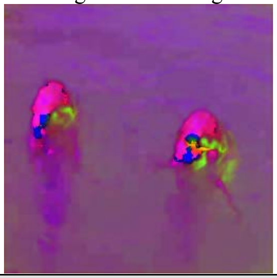
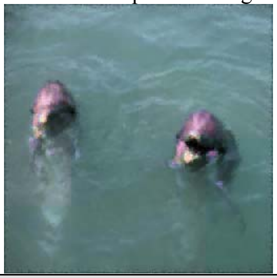
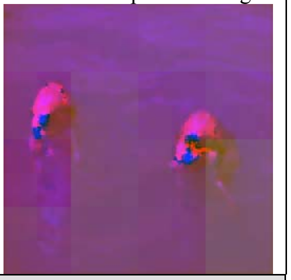
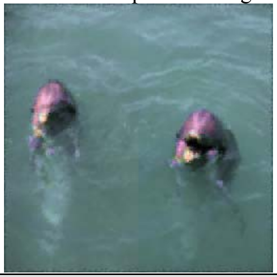
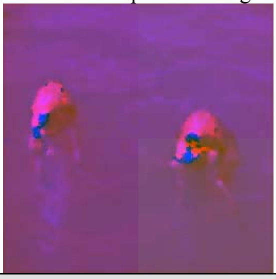
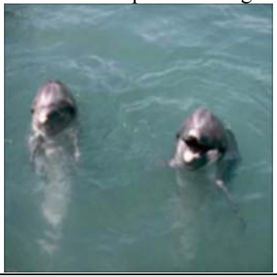
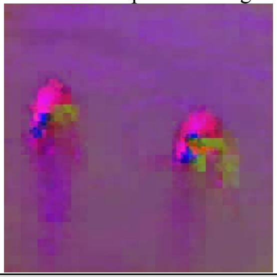
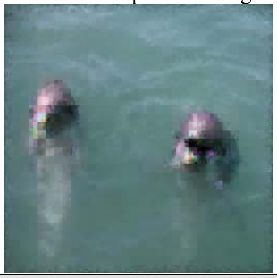
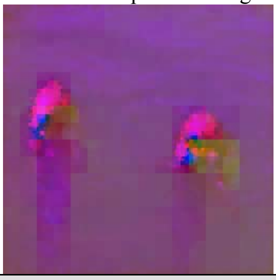
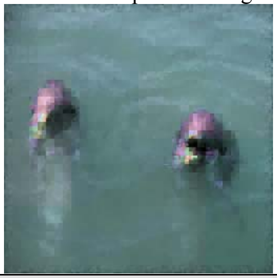
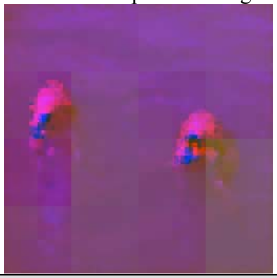
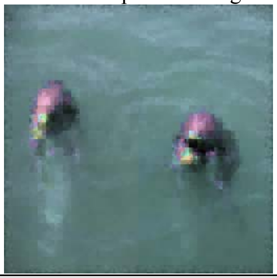
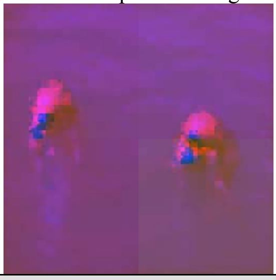
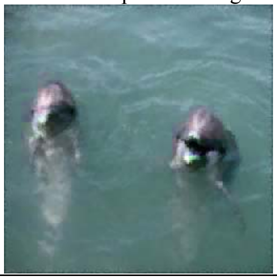
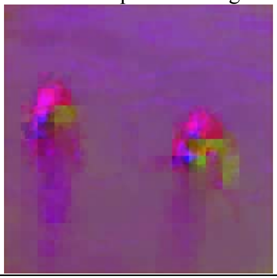
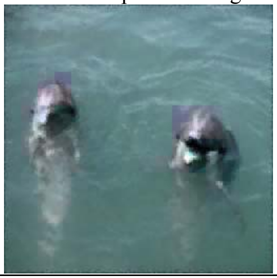
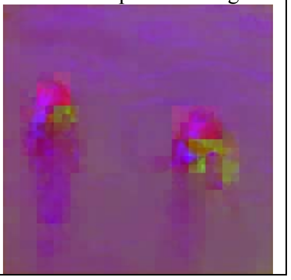
Image “18”			
Original Images		Size of the range blocks: H = 4, S = 64, and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4, S = 128, and V = 4		Size of the range blocks: H = 8, S = 16, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 8, S = 32, and V = 8		Size of the range blocks: H = 8, S = 64, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 8, S = 128, and V = 8		Size of the range blocks: H = 16, S = 16, and V = 4	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 32, S = 16, and V = 4			
RGB Decompressed Image	HSV Decompressed Image		
			

Figure B.18: Applying Three-Channel-HSV-Model to image "18"

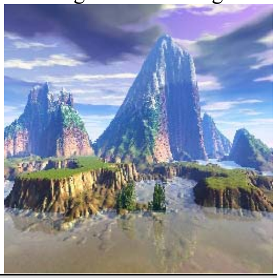
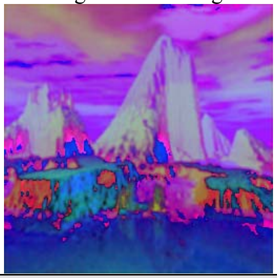
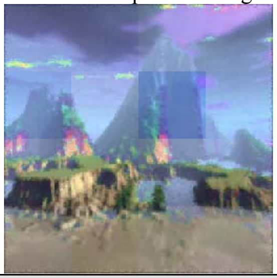
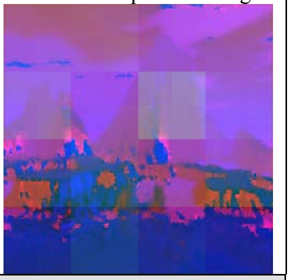
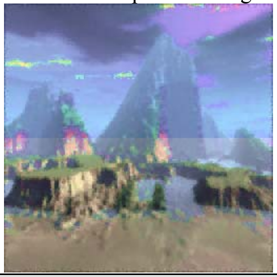
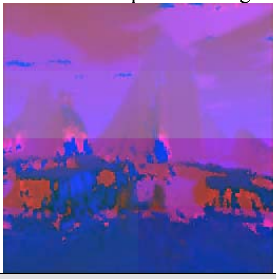
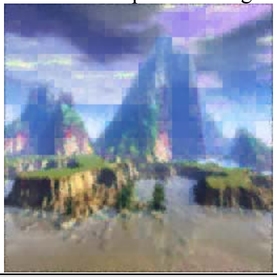
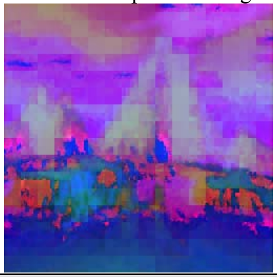
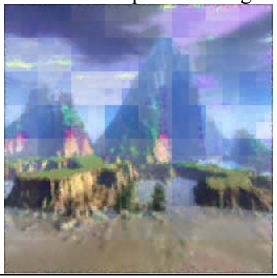
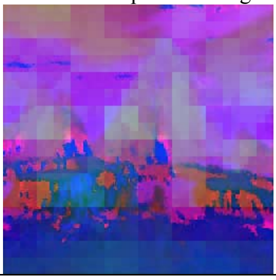
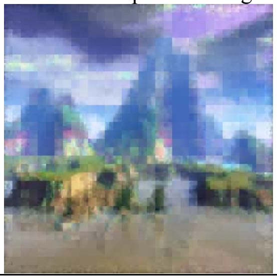
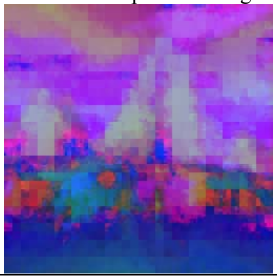
Image “19”			
Original Images		Size of the range blocks: H = 4, S = 64, and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4, S =128 , and V = 4		Size of the range blocks: H = 4, S = 16, and V = 4	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4, S = 32, and V = 4		Size of the range blocks: H = 8, S = 16, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			

Figure B. 19: Applying Three-Channel-HSV-Model to image "19"

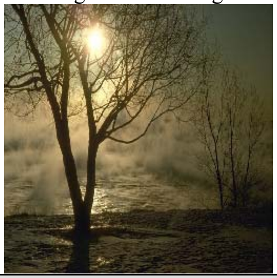
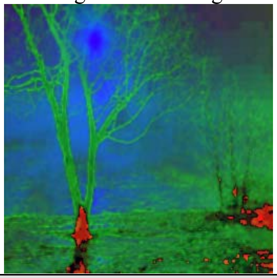
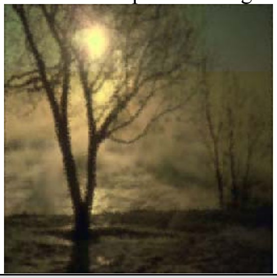
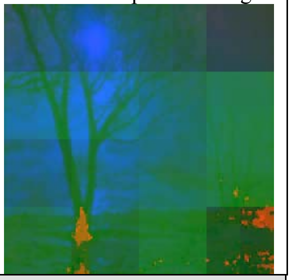
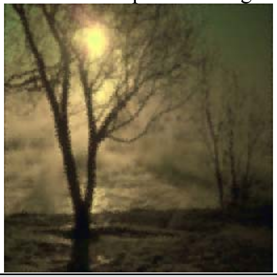
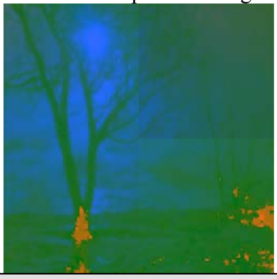
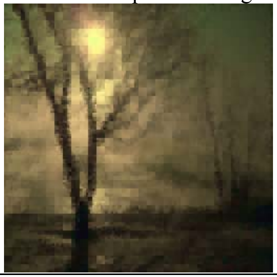
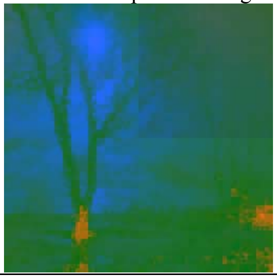
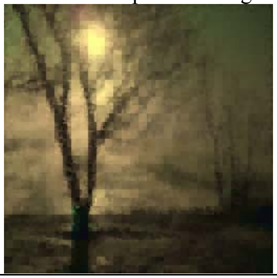
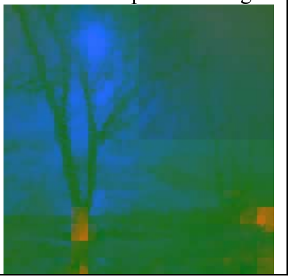
Image “20”			
Original Images		Size of the range blocks: H = 4, S = 64, and V = 4	
Original RGB Image	Original HSV Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 4, S = 128, and V = 4		Size of the range blocks: H = 8, S = 128, and V = 8	
RGB Decompressed Image	HSV Decompressed Image	RGB Decompressed Image	HSV Decompressed Image
			
Size of the range blocks: H = 16, S = 128, and V = 8			
RGB Decompressed Image	HSV Decompressed Image		
			

Figure B.20: Applying Three-Channel-HSV-Model to image "20"

Table B.1: The results of finding the Correlation, RMSE, and PSNR between original images (1-20) and their reconstructed images after applying the Three-Channel-HSV-Model.

H bits	S bits	V bits	Red Correlation	Green Correlation	Blue Correlation	Red RMSE	Green RMSE	Blue RMSE	Total RMSE	Red PSNR	Green PSNR	Blue PSNR	Total PSNR
Image "1"													
4	32	4	0.8846	0.8846	0.8966	32	31	32	95	18	18	18	54
4	64	4	0.8834	0.8997	0.8929	33	31	33	97	18	18	18	54
8	64	8	0.8436	0.8576	0.8559	37	37	38	112	17	17	17	51
16	64	8	0.8426	0.8565	0.8510	37	37	39	113	17	17	16	50
32	64	8	0.8419	0.8558	0.8501	38	37	40	115	17	17	16	50
64	64	8	0.8399	0.8565	0.8489	38	37	41	116	17	17	16	50
Image "2"													
4	32	4	0.9586	0.9358	0.9372	22	21	21	64	21	22	22	65
4	64	4	0.9575	0.9318	0.9343	22	21	21	64	21	22	22	65
8	64	8	0.9258	0.8990	0.9040	29	26	26	81	19	20	20	59
16	64	8	0.9215	0.9029	0.8977	31	26	27	84	18	20	20	58
32	64	8	0.9219	0.9058	0.8944	32	28	27	87	18	19	20	57
64	64	8	0.9277	0.9144	0.8921	33	30	27	90	18	19	20	57
Image "3"													
4	32	4	0.7850	0.8059	0.6513	34	33	34	101	18	18	18	54
4	64	4	0.7730	0.8035	0.5771	35	33	36	210	17	18	17	52
8	64	4	0.7680	0.8047	0.5667	36	33	37	212	17	18	17	52
16	64	4	0.7627	0.8024	0.5509	36	33	37	212	17	18	17	52
32	64	4	0.7657	0.8004	0.5422	36	33	37	106	17	18	17	52
Image "4"													
4	32	4	0.9630	0.9657	0.9719	14	13	11	38	25	26	27	78
4	64	4	0.9514	0.9590	0.9688	16	14	12	42	24	25	27	76
8	64	8	0.9417	0.9503	0.9611	18	15	13	46	23	25	26	74
8	64	16	0.9290	0.9366	0.9447	20	17	16	53	22	24	24	70
16	64	8	0.9400	0.9481	0.9630	18	16	13	47	23	24	26	73
32	64	8	0.9394	0.9445	0.9621	18	16	13	47	23	24	26	73

Table B.1: (Continued)

H bits	S bits	V bits	Red Correlation	Green Correlation	Blue Correlation	Red RMSE	Green RMSE	Blue RMSE	Total RMSE	Red PSNR	Green PSNR	Blue PSNR	Total PSNR
Image “4” (Continued)													
64	64	8	0.9395	0.9427	0.9620	18	17	13	48	23	24	26	73
64	64	16	0.9273	0.9274	0.9431	20	18	16	54	22	24	24	70
Image "5"													
4	64	4	0.9199	0.9313	0.8930	28	30	36	94	19	19	17	55
4	64	16	0.8507	0.8882	0.8536	37	38	42	117	17	17	16	50
8	64	8	0.8822	0.8981	0.8616	35	37	41	113	17	17	16	50
8	64	16	0.8423	0.8692	0.8325	39	41	45	125	16	16	15	47
16	64	8	0.8463	0.8819	0.8371	41	40	45	126	16	16	15	47
32	64	4	0.8268	0.8698	0.8344	48	43	47	138	15	15	15	45
Image "6"													
4	64	4	0.9002	0.9348	0.9557	28	26	25	79	19	20	20	59
4	128	4	0.8800	0.9310	0.9546	30	26	25	81	19	20	20	59
8	128	8	0.8425	0.9084	0.9428	34	31	28	93	18	18	19	55
Image "7"													
4	64	4	0.8754	0.8446	0.7605	34	32	33	99	18	18	18	54
4	128	4	0.8635	0.8439	0.7446	35	32	35	102	17	18	17	52
8	128	8	0.8439	0.8202	0.7245	38	35	36	109	17	17	17	51
8	128	16	0.8227	0.7893	0.7017	40	37	37	114	16	17	17	50
16	128	8	0.8407	0.8206	0.7225	38	34	36	108	17	18	17	52
32	128	8	0.8354	0.8180	0.7195	39	35	36	110	16	17	17	50
Image "8"													
4	64	4	0.8092	0.8347	0.6384	27	22	26	75	20	21	20	61
4	128	4	0.8081	0.8346	0.6356	27	22	26	75	20	21	20	61
8	128	8	0.6929	0.7141	0.4899	32	28	28	88	18	19	19	56
8	128	16	0.5836	0.5482	0.1642	36	33	32	101	17	18	18	53
16	128	8	0.6727	0.7133	0.5384	34	28	28	90	18	19	19	56

Table B.1: (Continued)

H bits	S bits	V bits	Red Correlation	Green Correlation	Blue Correlation	Red RMSE	Green RMSE	Blue RMSE	Total RMSE	Red PSNR	Green PSNR	Blue PSNR	Total PSNR
Image "9"													
4	64	4	0.9524	0.9514	0.9308	21	22	26	69	22	21	20	63
4	128	4	0.9518	0.9462	0.9293	21	23	27	71	22	21	20	63
8	128	8	0.9134	0.9166	0.9053	27	28	30	85	20	19	19	58
Image "10"													
4	64	4	0.9328	0.8524	0.6181	23	23	26	72	21	21	20	62
4	128	4	0.9326	0.8495	0.5865	23	23	27	73	21	21	20	62
8	128	8	0.8996	0.8158	0.5715	28	25	28	81	19	20	19	58
Image "11"													
4	64	4	0.8097	0.8899	0.9206	31	17	13	61	18	24	26	68
4	128	4	0.7289	0.8534	0.9132	37	19	14	70	17	23	25	65
8	128	8	0.7075	0.8298	0.8898	38	21	16	75	17	22	24	63
16	128	8	0.6960	0.8271	0.9026	38	21	15	74	17	22	25	64
16	128	16	0.6681	0.7716	0.8565	39	23	17	79	16	21	24	61
32	128	8	0.6969	0.8261	0.9111	38	21	14	73	17	22	25	64
Image "12"													
4	64	4	0.9095	0.9278	0.8667	29	28	39	96	19	19	16	54
4	128	4	0.9097	0.9220	0.8380	29	30	43	102	19	19	15	53
8	128	8	0.8325	0.8784	0.7878	39	37	46	122	16	17	15	48
16	128	8	0.8131	0.8635	0.7726	41	39	47	127	16	16	15	47
16	128	16	0.6719	0.8004	0.6774	52	46	53	151	14	15	14	43
32	128	8	0.7436	0.7904	0.7403	51	49	49	149	14	14	14	42

Table B.1: (Continued)

H bits	S bits	V bits	Red Correlation	Green Correlation	Blue Correlation	Red RMSE	Green RMSE	Blue RMSE	Total RMSE	Red PSNR	Green PSNR	Blue PSNR	Total PSNR
Image "13"													
4	64	4	0.9373	0.9520	0.8937	23	20	25	68	21	22	20	63
4	128	4	0.9364	0.9517	0.8935	23	20	25	68	21	22	20	63
8	128	8	0.8992	0.9288	0.8522	28	24	29	81	19	21	19	59
Image "14"													
4	64	4	0.9314	0.7965	0.4804	26	27	28	81	20	20	19	59
4	128	4	0.9316	0.7934	0.4426	26	27	28	81	20	20	19	59
8	128	8	0.8850	0.6936	0.3806	34	32	30	96	18	18	19	55
Image "15"													
32	128	8	0.8158	0.8388	0.8131	38	34	35	107	17	18	17	52
4	64	4	0.8856	0.8914	0.8827	29	28	28	85	19	19	19	57
4	128	4	0.8727	0.8857	0.8746	32	29	29	90	18	19	19	56
8	128	8	0.8211	0.8431	0.8273	37	33	34	104	17	18	18	53
16	128	8	0.8185	0.8425	0.8211	37	33	34	104	17	18	18	53
Image "16"													
4	64	4	0.8511	0.8567	0.8228	32	31	29	92	18	18	19	55
4	128	4	0.8419	0.8506	0.8163	33	31	30	94	18	18	19	55
8	128	8	0.7955	0.8055	0.7799	36	35	32	103	17	17	18	52
16	16	4	0.8630	0.8672	0.8458	30	29	27	86	19	19	20	58
16	64	16	0.7579	0.7572	0.7450	39	39	34	112	16	16	18	50
16	128	8	0.7919	0.8064	0.7819	37	35	32	104	17	17	18	52
16	128	16	0.7502	0.7552	0.7414	40	39	34	113	16	16	18	50

Table B.1: (Continued)

H bits	S bits	V bits	Red Correlation	Green Correlation	Blue Correlation	Red RMSE	Green RMSE	Blue RMSE	Total RMSE	Red PSNR	Green PSNR	Blue PSNR	Total PSNR
Image "17"													
4	64	4	0.9164	0.9097	0.8970	29	26	26	81	19	20	20	59
4	128	4	0.9052	0.9024	0.8924	34	28	26	88	18	19	20	57
8	64	8	0.8738	0.8600	0.8407	35	32	31	98	17	18	18	53
Image "18"													
4	64	4	0.9300	0.9254	0.9468	10	10	9	29	28	28	29	85
4	128	4	0.9183	0.9153	0.9416	10	10	9	29	28	28	29	85
8	16	8	0.9199	0.9149	0.9285	10	10	10	30	28	28	28	84
8	32	8	0.9122	0.9105	0.9275	11	10	10	31	27	28	28	83
8	64	8	0.8990	0.8991	0.9248	12	11	10	33	27	27	28	82
8	128	8	0.8821	0.8904	0.9194	13	11	11	35	26	27	27	80
16	16	4	0.9409	0.9399	0.9515	9	9	8	26	29	29	30	88
32	16	4	0.9407	0.9361	0.9513	9	9	8	26	29	29	30	88
Image "19"													
4	64	4	0.7801	0.8272	0.9427	32	25	18	75	18	20	23	61
4	128	4	0.7655	0.8154	0.9387	33	26	19	78	18	20	23	61
4	16	4	0.8894	0.8966	0.9567	23	20	16	59	21	22	24	67
4	32	4	0.8499	0.8679	0.9512	27	22	17	66	20	21	24	65
8	16	8	0.8560	0.8652	0.9509	26	22	17	65	20	21	24	65
Image "20"													
4	64	4	0.9444	0.9315	0.8699	18	18	20	56	23	23	22	68
4	128	4	0.9436	0.9307	0.8588	18	18	20	56	23	23	22	68
8	128	8	0.9191	0.9029	0.8260	21	21	22	64	22	22	21	65
16	128	8	0.9204	0.9035	0.8279	21	21	22	64	22	22	21	65

APPENDIX –C

SAMPLE CODE

The simulation was done using Matlab 6.5 software.

```
**********************************************************************
% Code sample for the Three-Channel-HSV-Model:

function [RGB_APP,PSNR,RMSE,tim] = Model6 (RGB,Size,IP,H_RangeSize,S_RangeSize,V_RangeSize)

cd('D:\MATLAB6p5\work\HSI Fractal Compression')

RGB = imresize(RGB,Size);

h = fspecial('average',[3 3]);
RGB = imfilter(RGB,h);

HSV = rgb2hsv(RGB);

t = cputime  ;

F_Image_H = F_Comp((HSV(:,:,1)*255),H_RangeSize);
F_Image_S = F_Comp((HSV(:,:,2)*255),S_RangeSize);
F_Image_V = F_Comp((HSV(:,:,3)*255),V_RangeSize);

tim = cputime-t;

H_DF = Decompression(IP,Size, F_Image_H,H_RangeSize);
H_DF = double(uint8(H_DF))/255;

S_DF = Decompression(IP,Size, F_Image_S,S_RangeSize);
S_DF = double(uint8(S_DF))/255;

V_DF = Decompression(IP,Size, F_Image_V,V_RangeSize);
V_DF = double(uint8(V_DF))/255;

HSV_APP(:,:,1) = H_DF;
HSV_APP(:,:,2) = S_DF;
HSV_APP(:,:,3) = V_DF;

RGB_APP = hsv2rgb(double(HSV_APP));
RGB_APP = RGB_APP*255;

RMSE = Root_Mean_SQR_Error(double(RGB) , RGB_APP)
PSNR = 20*log10(255/RMSE)
cd('D:\MATLAB6p5\work\Model6')

**********************************************************************
% Reduced RGB Model:
function Reduced_RGB(Image,ImageSize,Size_Range_Block,init)
II = imresize(Image,ImageSize);
I = II;
%Image Blurring
h = fspecial('average',[3 3]);
I = imfilter(I,h);
cd('D:\MATLAB6p5\work\Quantization')
Image = color_compression(I,ImageSize(1));
cd('D:\MATLAB6p5\work\HSI Fractal Compression')
```

```
T = F_Comp(Image,Size_Range_Block);
init = imresize(init,ImageSize);
DF = Decompression(init,ImageSize, T,Size_Range_Block);

cd('D:\MATLAB6p5\work\Quantization')
Decompressed_Image = color_decompression (double(uint8(DF)),ImageSize(1));
cd('D:\MATLAB6p5\work\ReducedRGB')

*****************************************************************************************
% Reduced HSV Model

function Reduced_HSV(Image,ImageSize,Size_Range_Block,init)

II = imresize(Image,ImageSize);
I = II;

%Image Blurring
h = fspecial('average',[3 3]);
I = imfilter(I,h);

%conversion to HSV color space
HSV = rgb2hsv(I);

cd('D:\MATLAB6p5\work\Quantization')
Image = HSI_compression(HSV,ImageSize(1),1);

cd('D:\MATLAB6p5\work\HSI Fractal Compression')
T = F_Comp(Image,Size_Range_Block);

init = imresize(init,ImageSize);
DF = Decompression(init,ImageSize, T,Size_Range_Block);


cd('D:\MATLAB6p5\work\Quantization')
Decompressed_Image = HSI_decompression (double(uint8(DF)),ImageSize(1),1);

cd('D:\MATLAB6p5\work\ReducedHSV')
*****************************************************************************************
```

ضـغـط الـصـور الـمـلونـة بـالـطـريـقـة الـتـجـزيـئـيـة

إعـداد
روان إسماعيل زغلول

المشـرف
الدكتور سـامي سـرحان

الملخـــــــص

إن الهدف الرئيسي من ضغط الصور بالطريقة التجزيئية هو البحث عن الأجزاء المتـشابهة في الصورة و تحويلها إلى صيغ رياضية سهلة التمثيل، وبأقل سعة تخزين ممكنة.و لما كانت العديد من الأبحاث المتعلقة بضغط الصور التجزيئية موجهةً نحو ضغط الصور غيـر الملونـة، كانـت الأبحاث لضغط الصور الملونة بالطريقة التجزيئية قليلة جداً. وحتى يكون موضوع البحث متكـاملاً يجب أن يكون لدينا رؤيا واضحة تشمل جميع جوانب البحث، ومن هنا كانت بداية التفكير في ضغط الصور الملونة بالطريقة التجزيئية ليكون موضوعاً لهذه الدراسة.

في هذه الدراسة قمنا باقتراح عدة أنظمة، معظمها اعتمد على الطريقـة البيولوجيـة لـدى الانسان في تعرف الألوان، وذلك عن طريق استخدام الفضاء اللوني المسمى بالـ HSV. فقـد تـم اقتراح ستة أنظمة وتحليلها، أول خمسة منها لم تحقق النتائج المرجوة سواء من ناحية معدل ضـغط الصورة أو نوعية الصورة المضغوطة من ناحية أخرى؛ ومن هنا كانت الحاجة لتطوير نظام جديـد يجمع بين معدل الضغط المرتفع والنوعية الجيدة للصورة المضغوطة. لذلك قمنا بتطـوير النظـام

الأخير الذي يدعى بنموذج الـ HSV ثلاثي القنوات ذي المشفرات التجزيئية المختلفـة. باسـتخدام هذا النموذج توصلنا الى معدل ضغط مرتفع (ما يقارب الـ 60:1). و من أجل أن تكون دراسـتنا شاملة لجميع جوانب الموضوع كان من الواجب علينا أن نقوم بمقارنة نتائجنا مع ما توصـل اليـه باحثون آخرون في نفس المجال.

إن النتائج التجريبية لهذه الدراسة تقدم أدوات وتوجيهات واضحة المعالم لـضغط الـصور الملونة بالطريقة التجزيئية. وبالمقارنة مع الطرق السابقة ، حقق نموذجنـا معـدل ضـغط مرتفـع للصور المختارة وسرعة أداء جيدة في معظم أنواع الصور خاصة التي تتميزبنمط من التشابه بـين أجزاءها دون تغيرات حادة في ألوانها.