Classification and Compression Based on Component Analysis of Images

A Thesis

Submitted to the College of Graduate Studies and Research

in Partial Fulfillment of the Requirements

for the Degree of

Master of Science

in the

Department of Electrical Engineering

University of Saskatchewan

Saskatoon, Saskatchewan

By

Li Yuan

August 1997

Permission to Use

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Electrical Engineering
University of Saskatchewan
Saskatoon, Saskatchewan
Canada S7N 5A9

UNIVERSITY OF SASKATCHEWAN
Electrical Engineering Abstract 97A462

Abstract

The application of visual pattern recognition in agriculture is a growing field of study. Accurate objective inspection and grading of agriculture products using a computer vision system hold great potential for product quality control and sorting. In this thesis, colour features of barley seeds are used as feature inputs to a multi-layer neural network in barley seed variety classification experiments. Four barley varieties, Tankard, Breedn, Creme, and CDC Dolly, were tested. 140 samples for each variety were used for pattern recognition training and an additional 140 samples for testing. Different colour spaces RGB, Lab, and HSI were used to represent colour features. The results showed that HSI colour features were the best for variety classification. The best recognition accuracy for individual seeds was 82.7% using a neural network with a structure of 4 inputs, 20 neurons in each of two hidden layers, and 4 outputs.

With the increasing need to store and electronically transmit images, the study of image coding is getting more attention. A major objective of lossless image coding is to represent an image with as few bits as possible without loss of any information in the image. In this thesis, the statistical properties of an image were studied to compare lossless image data compression schemes. Huffman coding is reviewed, and two methods of constructing a difference image were implemented and evaluated. After applying these two methods to MRI images, compression ratios were calculated. The standard difference image had the lowest compression ratio. The advantage of an hierarchy embedded differential image is that the image quality can be progressively improved as the image transmission progresses. A coding method which considered the correlation of the adjacent pixels was also studied. The compression results by using this method on MRI images were discussed.

Acknowledgements

I would like to express my sincere thanks to Dr. K. Takaya and Dr. H.C. Wood for their supervision and advice during the course of this project. I also express thanks to Maurice D. Romaniuk for his help in the seed classification experiments.

I gratefully acknowledge the financial support of the TRlabs in the form of a scholarship.

I would like to thank my parents and my sister for their unlimited love. Most importantly, I would like to thank my husband, Jun Mu, and my son, Yuqiang Mu, for their generous love and support. Without their understanding and support, this thesis would not have been written.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 General

Due to the increasing utilization of imagery in many applications, coupled with improvements in the size, speed, and cost effectiveness of digital computers and related signal processing technologies, the field of digital image processing has grown considerably during the past decade. Image processing technology has been applied in many areas.

Digital image processing can be classified broadly into four areas: image enhancement, restoration, coding, and analysis. Image analysis and image coding are two especially important fields in image processing; applications of image analysis include computer vision and robotics. In industry, for example, an artificial vision system can

provide an objective analysis of parameters such as shape or colour and can eliminate human factors caused by fatigue and stress which influence an inspection process. In the communication field, on the other hand, due to the advances in multi-media applications and digital imaging technology, obtaining high resolution digital images from databases over a computer network is becoming more and more popular. The need for compressing images, both for archival and for transmission purposes, has been growing.

The projects considered in this thesis cover two important aspects of image processing, image classification and image compression; one is sensitive to the contents of an image and the other is independent of the contents.

1.2 Research Objectives

1.2.1 The Visual Pattern Recognition Problem

The computer vision branch of artificial intelligence is concerned with developing algorithms for analyzing image content. The visual pattern recognition approach assumes that the image may contain one or more objects and that each object belongs to one of several predetermined types or classes. Objects are identified using image analysis methods that measure the distinguishing attributes of these objects and match them to those of the predetermined types. One fundamental goal of visual

pattern recognition is to improve the speed and accuracy of the process by emulating the visual information gathering and processing capabilities of the human eye and brain. This goal can usually be achieved by combining image analysis with pattern recognition algorithms.

Most optical vision systems create two dimensional images of objects. Objects in the field of view of a camera are identified using image analysis techniques that determine important features unique to those objects. A *pattern space* is defined as a set of those measurable features which distinguish one kind of object from another. To classify different classes, decision boundaries, which separate the class feature clusters, must be established using pattern recognition algorithms. To achieve accurate pattern recognition, it is very important to choose proper features which can adequately represent objects. A number of attributes of objects could be used as features, such as object size, area, shape attributes, and colour. For many applications, colour is a very important visual attribute of objects.

The application of visual pattern recognition in agriculture has been increasing rapidly. Accurate objective inspection and grading of agricultural products using a computer vision system hold great potential for product quality control and sorting. Agricultural product inspection and grading traditionally depend on human judgment; these evaluations are very laborious, repetitive, and subjective, especially when colour is used as one of the features. The perception of colour by the human

eye is subject to individual differences among inspectors; it also depends on the state of visual adaptation of an individual. In contrast, machine vision systems can be designed to be objective and reproducible.

Visual pattern recognition based on the colour of agricultural products encompasses a variety of applications, such as wheat class discrimination based on colour analysis [11] [12], colour classification of core germplasm [14], colour analysis of peaches for grading [10], modeling sensory colour quality of tomatoes and peaches [21], discrimination of mustard and canola seeds [4], and recognition of barley seeds based on their shape [18].

Seed inspection and grading is an important research area in machine vision systems for agriculture. Different kinds of barley seeds could be determined based on colour and shape information. Colour is one of the most important features, since seed maturity and seed quality are highly correlated with the seed colour. Representing barley seeds in terms of their colour information for automated variety discrimination is one of the research aspects in this work.

There are numerous pattern recognition algorithms given in the literature, such as those in [9] [3] [18]. Neural network approaches to visual pattern recognition have been used increasingly in recent years. Neural networks attempt to achieve human-like performance and to perform as a classifier which has both adaptive and robust capabilities [5]. In this research, a multi-layer neural network is used as a pattern

recognition classifier. Testing recognition accuracy using a neural network is a goal of this research.

1.2.2 The Image Compression Problem

Despite rapid gains in storage and data transmission technology, there is an increasing need for image compression. This need is due to the development of new image modalities, such as MRI, and the digitization of conventional images, which generates vast amounts of data. There is also increasing demand for transmission of these data over computer networks.

The overall goal of compression is to represent an image with the smallest possible number of bits, thereby reducing transmission time and minimizing storage requirements. There are two basic kinds of image compression: lossless and lossy. Lossless compression is fully reversible, with no loss of information, but is typically limited to low compression ratios. Lossy compression is irreversible and loses information, but usually results in quite high compression ratios. In some cases, any loss of information is not allowed, especially in medical applications. Doctors and physicians prefer to work with the exact images obtained from a measurement apparatus. In these cases, lossless algorithms are used for compression.

Magnetic Resonance Imaging (MRI) is a relatively new imaging technique, which

has become an essential part of diagnostic radiology over the last decade. One important advantage of MRI is its high quality resolution of soft tissues. Due to this advantage, MRI has become the most comprehensive method of diagnosing and monitoring diseases of the brain. Megnetic Resonance Imaging is also used in some agricultural applications to examine the interior of seeds or fruits.

The size of MR image data files is quite large. A set of MRI data (64 slices) of a human head is typically around 10 Megabytes. It is necessary to do image compression to minimize storage requirements, and to speed transmission over computer networks.

Many types of algorithms exist in the image compression field. One of the objectives of this thesis is to investigate some image compression algorithms to try to obtain higher compression ratios without loss of any information. Those algorithms are developed based on image statistics. The algorithms are evaluated by using them for 2D MRI images, and the well known image "lena". One of the commonly used image coding methods, Huffman coding, is used to code the images and the image compression ratios are calculated.

The primary objectives of this thesis are to examine the usefulness of colour as a classification parameter for seeds, and to examine coding techniques to perform lossless compression of MRI images for efficient storage and transmission.

1.3 Outline of Thesis

This thesis is arranged into five chapters and one appendix. The first chapter addresses the aspects of image analysis, pattern recognition and image coding. The thesis objectives and outline are also introduced.

The second chapter consists of three parts. The first part introduces some basic concepts about digital colour images and colour spaces. The second part describes some image analysis techniques which can be used for seed classification. The histogram threshold and border following methods are introduced for object isolation. The KL transform technique for evaluating feature selection is introduced. The principle of a multi-layer neural network as a classifier is described. The third part is about image coding. In this part, procedures for image coding are introduced. The statistics of an image are presented for the development of an efficient data compression technique. Some difference image construction methods are introduced and the results analyzed.

The third chapter concentrates on barley seed classification using image analysis. The machine vision system setup for image analysis is introduced first. The image capturing and feature extraction of barley seeds are described. The testing and training processes of a multi-layer neural network as a classifier are described. The results of different experiments are presented in the last part of this chapter.

The forth chapter deals with the lossless image coding problem. Two algorithms

for constructing a difference image are introduced and the evaluation of these algorithms, based on information entropy, is done by applying these methods to a magnetic resonance (MR) image. Because of availability, MR images of a human head are used for the analysis, rather than images of a barley seed. The principles are the same. A modified Huffman coding method is described and implemented. Experimental results for compressing MR images are shown.

The fifth chapter includes a brief summary of the results within this thesis and the conclusions.

Chapter 2

Image Analysis and Image Coding

2.1 Introduction

Image analysis and image coding are two important branches of image processing. Image analysis is concerned with the extraction of useful information from an acquired image. Image coding involves methods for representing monochrome and colour images with a minimal number of code words for more efficient communication and storage.

In image analysis, the input is an image, while the output is typically some symbolic representation of the contents of the input image. Successful development of systems in this area involves both signal processing and artificial intelligence concepts. It involves first finding the objects within an image and then identifying (classifying)

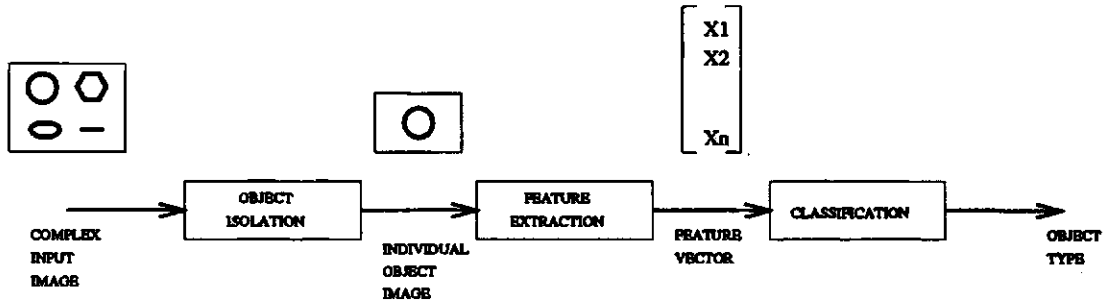


Figure 2.1: The three phases of pattern recognition (modified from[12])

those objects according to the extracted features of the image.

Among many approaches of "image analysis", we only focus in this work on visual pattern recognition. This approach assumes that the image may contain one or several predetermined types or classes.

Given a digitized image containing several objects, the pattern recognition process consists of three major phases: object isolation, feature extraction, and classification (as shown in Figure 2.1). In the first phase, object isolation, each object must be found and its image must be isolated from the rest of the scene. In the second phase, feature extraction, a set of measurable properties, called the measurement vector, is determined. This measurement vector could be used as a feature vector directly or transferred to another proper vector as a feature vector. In the third phase, object classification, a decision regarding the class to which the object belongs is made. i.e. each object is assigned to one of several pre-established classes. The classification is based solely on the feature vector.

A major objective of image coding is to represent an image with as few bits as possible while preserving the level of quality required for a given application. Image coding is divided into two classes according to the quality of the reconstructed image: *lossy coding* and *lossless coding*. Image coding techniques which do not allow any loss of information are referred to as *lossless coding*, also called *information preserving coding*. On the other hand, image coding techniques which allow some information loss are called *information lossy* or *lossy coding*. Lossy coding techniques can reduce the volume of data rather drastically at the expense of image quality. Usually, it can provide an order of magnitude higher compression ratio than that of lossless coding [6]. However, some applications do not tolerate any loss of information when images are retrieved or accessed over a network for the purpose of viewing or analysis, such as in medical image applications. In this thesis, lossless image coding is studied.

Usually, image coding is accomplished by a three-stage process: transformation, quantization, and codeword assignment. The first stage is an important stage in which the image is transformed into a more condensed form. This step determines what will be coded. Image coding is classified into three categories according to which aspect of an image is to be coded. One class is called *source coding*, which codes the image intensity or some variation of image intensity without transforming to another image. Another class is called *transform coding*, which encodes some transform coefficients

of an image, such as Fourier transform coefficients. The third one is called *image model coding*, in which an image or some portion of an image is modeled and the model parameters are coded. As an example, fractal base image coding extracts a basic fractal pattern as a model and the parameters describing this model are coded.

The second stage of coding is quantization. In this stage, source data, transform coefficients, or model parameters of the image are quantized with the help of a compandor so an image can be adequately presented numerically without losing the details of the image. In the third stage, codeword assignment, strings of bits(codeword) are assigned to represent the quantization levels.

Efficiency is an important issue in evaluating a coding scheme. Usually, the compression ratio is used as a parameter to indicate coding efficiency. The *code rate* or *bit rate* of the compression is defined as the average number of bits produced per image pixel. The *compression ratio* (CR), is generally defined as

$$CR = \frac{\text{bit rate of original image}}{\text{bit rate of compressed image}} \tag{2.1}$$

If the original image has 8 bits per pixel (bpp) and the compression algorithm reduces it to a rate of 5 bpp, then compression ratio is 1.6. Keeping in mind that medical images need to be compressed as much as possible without losses, Source coding is attempted to find whether or not medical images can be compressed better than CR = 2, a commonly accepted compression barrier.

In this chapter, the basic concepts of image and colour space are introduced. The

Karhunen-Loéve (KL) transform technique, which is used as a reference standard of feature selection, is presented [9]. The threshold image segmentation technique and an eight-point border following algorithm for object connectivity are discussed. An artificial neural network, which is used as a pattern classifier, is described. The image statistics and Huffman coding are also introduced in this chapter.

2.2 Digital Image and Colour Space

An image is a two-dimensional representation of a real physical three-dimensional scene. A digital image is typically obtained by sampling an analogue image, for instance, an image on a film.

A point in the image plane has a colour which can be specified by the wavelength of light [13]. In the vast, continuous spectrum of electromagnetic energy, visible light is only within the wavelength band of 380 to 780 nanometers.

For a black-and-white image sensitive only to light intensity, brightness at a location (x, y) can be represented by

$$I(x,y) = k \int_{\lambda=0}^{\infty} c(\lambda)\, p(\lambda)\, s_{BW}[\lambda(x,y)]\, d\lambda \tag{2.2}$$

where $c(\lambda)$ is the spectral power distribution (in watts per square meter per unit wavelength) of the light, $p(\lambda)$ is the sensor spectral sensivity, $s_{BW}(\lambda)$ is the spectral reflectance of an object, and k is some scaling constant. Usually, in image processing

I is scaled to $0 \leq I \leq 255$ (8 bits) in digitized images. The value I is referred to as *graylevel* or *luminance*. Each level is commonly denoted by an integer, with 0 corresponding to the darkest level and 255 to the brightest. When an image plane is divided into small cells, the cell is called a pixel or pel (picture element).

A colour image can be considered as a combination of three monochrome images, and is described by three colour coordinates. These three coordinates can be set differently in the colour space, for example, RGB, HSI, and Lab. In colour image processing, one of such coordinate systems can be chosen to suit the purpose of the image processing. One of the frequently used colour coordinates is the RGB coordinate system, which presents the intensities of red, green, and blue components. A colour coordinate system can also be transformed to another colour coordinate system by a transformation matrix.

RGB space

Almost all perceived colours can be produced by the combination of three primary colours [7]. Green, blue and red are chosen as primary colours of the additive colour system because,by proper combination, they can produce a wider range of colours than any other combination of three colours. According to the C.I.E.[1], $\lambda = 700$ nm has been chosen for red, $\lambda = 546.1$ nm has been chosen for green, and $\lambda = 435.8$ nm has been chosen for blue.

[1]C.I.E. stands by the Commission International de l'Eclairage.

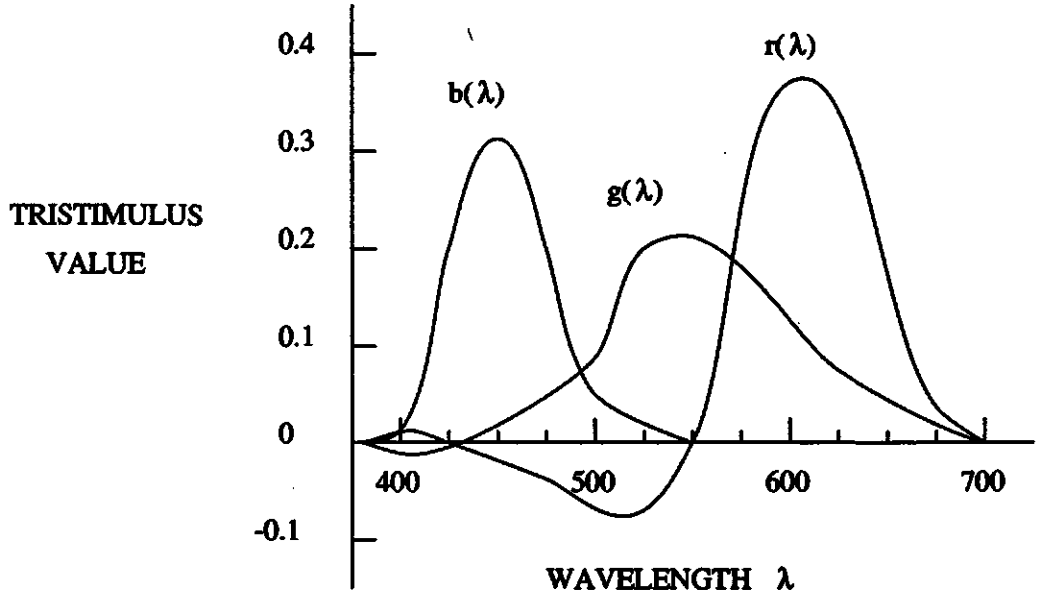


Figure 2.2: The colour matching functions for RGB space (from [15])

If the spectral energy distribution of light is $c(\lambda)$, then the colour of the light can be represented by integrating the appropriate colour matching function $r(\lambda)$, $g(\lambda)$, and $b(\lambda)$ as follows:

$$R = k \int_{\lambda=0}^{\infty} c(\lambda)\, r(\lambda)\, d\lambda \tag{2.3}$$

$$G = k \int_{\lambda=0}^{\infty} c(\lambda)\, g(\lambda)\, d\lambda \tag{2.4}$$

$$B = k \int_{\lambda=0}^{\infty} c(\lambda)\, b(\lambda)\, d\lambda \tag{2.5}$$

where the constant k scales R, G, and B values between 0 and 255. Fig 2.2 shows the colour matching functions for RGB space [13].

The National Television Systems Committee (N.T.S.C.) defined a similar RGB coordinate. The relationship between the C.I.E. spectral primary system and the NTSC. primary system is presented by a pair of linear coordinate conversions. This conversion can be found in Table 3.5-1 of [15].

XYZ space

The C.I.E. developed an artificial primary coordinate system in which all the tristimulus values in the matching functions are positive. The transformation from RGB space to XYZ space is given by:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.607 & 0.174 & 0.201 \\ 0.299 & 0.587 & 0.114 \\ 0.000 & 0.066 & 1.117 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2.6}$$

where R, G, and B are the values used under the NTSC colour system.

YIQ space

Human perception of light is generally described in terms of *brightness*, *hue* and *saturation*. *Hue* refers to the colour, and it normally corresponds to the stimulation of the retina by a narrow wavelength band of light. *Saturation*, or *chroma*, indicates how vivid or dull the colour is. A "pure" colour means that it is fully saturated.

Brightness indicates how bright the light is. Usually it is represented by the intensity of the light. YIQ space, denoted as *luminance – chrominance*, is one quite useful colour set in practice since it is more closely related to human perception than any other space. In the YIQ space, the Y component is called the luminance component. It is a measure of the luminance of a colour. I and Q components jointly describe the hue and saturation of the image.

The transformations from RGB space to YIQ space is given by ([15]):

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.523 & 0.312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2.7}$$

where R, G, and B are values used under the NTSC colour system.

HSI space

Another important colour space is the HSI space [20], where H denotes hue, S denotes saturation, and I denotes intensity. HSI colour space is shown as Fig 2.3, where intensity lies along the central axis, the distance from the axis gives the Saturation, and the direction specifies the Hue.

The transformations from RGB space to HSI space is given by [20]:

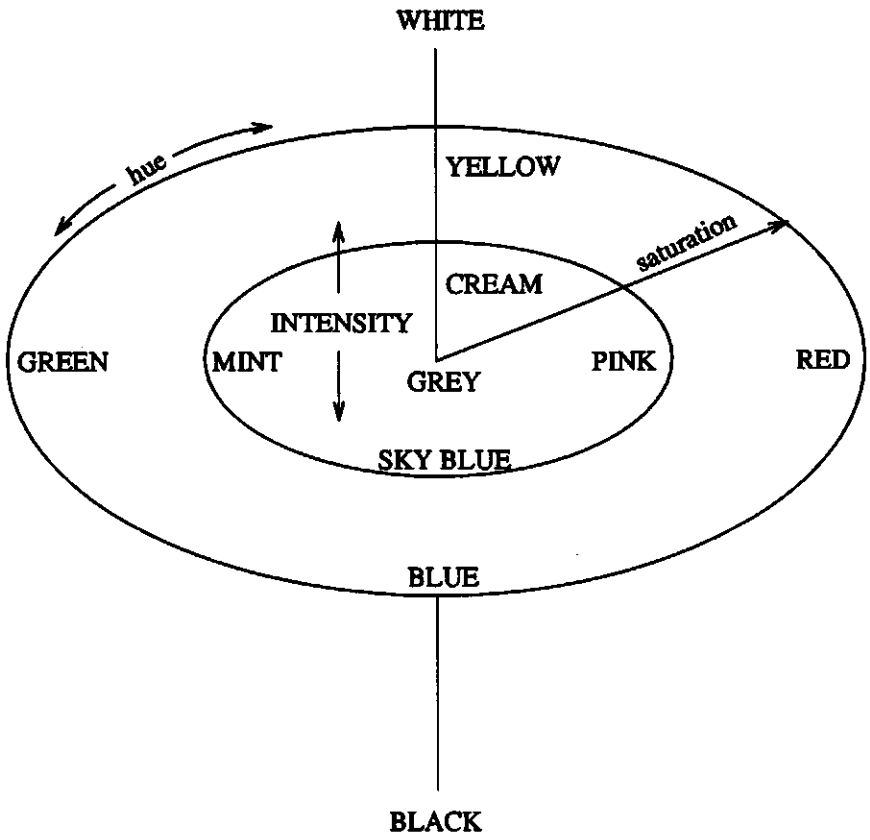


Figure 2.3: HSI colour space

$$H = \begin{cases} \left[\pi/2 - \arctan\left\{(2R-G-B)/\sqrt{3}\,(G-B)\right\} + \pi\right]/2\pi & \text{if } G < B \\ \left[\pi/2 - \arctan\left\{(2R-G-B)/\sqrt{3}\,(G-B)\right\}\right]/2\pi & \text{otherwise} \end{cases} \quad (2.8)$$

$$I = (R+G+B)/3 \quad (2.9)$$

$$S = 1 - [\min(R,G,B)/I] \quad (2.10)$$

Lab space

The Lab colour coordinate system is a uniform colour scaling system. It has been developed to provide a colour space which is uniform in the spacing of colours as related to visual difference. Basically, L is related to the brightness, a to redness-greenness, and b to yellowness-blueness. The system of L-a-b space is shown in Fig 2.4.

The L, a, and b values are given by:

$$L = 25\left(100\frac{Y}{Y_0}\right)^{\frac{1}{3}} - 16 \quad (2.11)$$

$$a = 500\left[\left(\frac{X}{X_0}\right)^{\frac{1}{3}} - \left(\frac{Y}{Y_0}\right)^{\frac{1}{3}}\right] \quad (2.12)$$

$$b = 200\left[\left(\frac{Y}{Y_0}\right)^{\frac{1}{3}} - \left(\frac{Z}{Z_0}\right)^{\frac{1}{3}}\right] \quad (2.13)$$

where X_0, Y_0, and Z_0 are the values of X, Y, and Z for the white reference, i.e., the values of X, Y, and Z when the R, G, and B values are all 255.

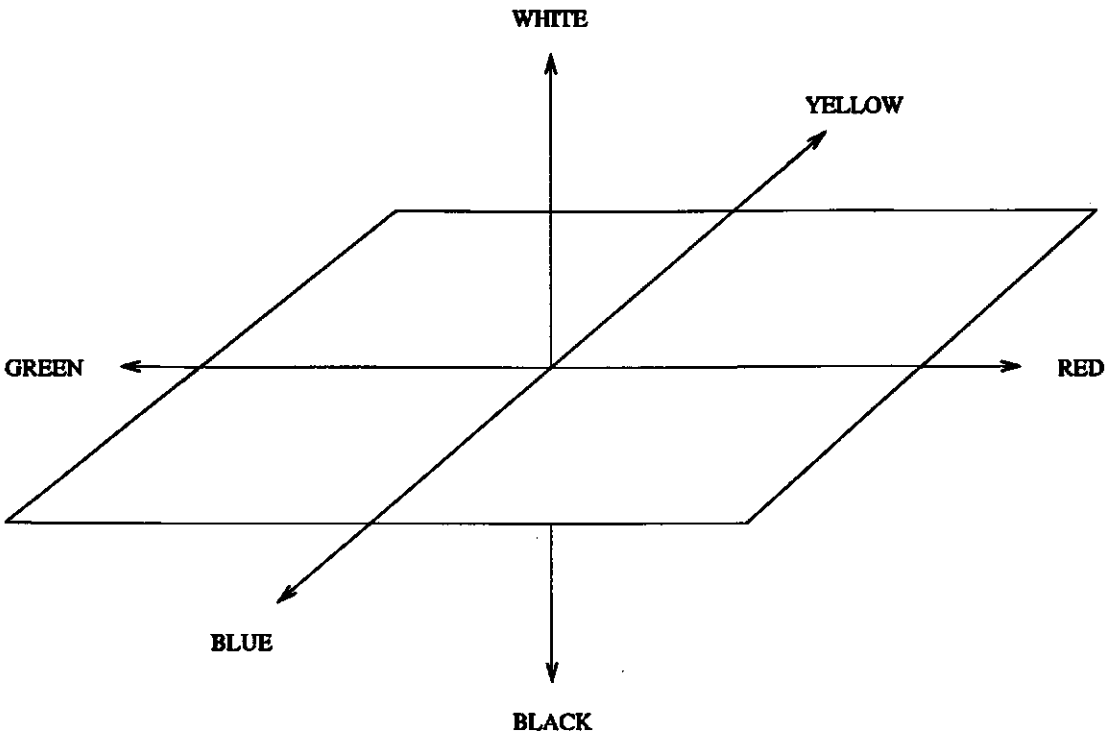


Figure 2.4: The Lab colour space

Colour space selection in image analysis

Using colour as a feature of the objects in image analysis is one of the objectives of this thesis. Various colour spaces have been used in barley seed classification, and several experiments have been done using different colour spaces. Based on the experimental results, the optimum colour space has been chosen. The Karhunen-Loéve transform is used as a reference to determine which colour coordinate system works best for recognizing colour based features.

2.3 KL Transform

The Karhunen-Loéve transform is an optimal orthogonal transform [9] in that the greatest amount of energy is packed into the fewest number of transform coefficients. The variables used to describe a signal or a set of data become uncorrelated after applying the KL transform, so one transform coefficient can be processed independently without affecting the others. The basis functions of the KL transform are the eigenvectors of the covariance matrix of a given sequence.

Let us denote a signal sequence as vector $\mathbf{b}$ of length N. The KL transformation matrix $\mathbf{T}$ is derived from the $N \times N$ covariance matrix $\mathbf{C}$ of the sequence $\mathbf{b}$:

$$\mathbf{C} = E\left[(\mathbf{b} - E[\mathbf{b}])(\mathbf{b} - E[\mathbf{b}])'\right] \tag{2.14}$$

or in terms of the elements of C and b:

$$c(i,j) = E\left[(b_i - E\left[b_i\right])\left(b_j - E\left[b_j\right]\right)\right], \tag{2.15}$$

where $E[b]$ denotes the expectation of b, and $'$ denotes the transpose operation.

The basis vectors of the KL transformation are orthonormalized eigenvectors of C, i.e.,

$$\mathbf{C}\mathbf{t}_m = \lambda_m \mathbf{t}_m \tag{2.16}$$

where $\{\lambda_m\}$ are the eigenvalues of C and $\{\mathbf{t_m}\}$ is the basis vector of the KL transform described by Eq 2.16.

Choose the eigenvectors corresponding to the n largest eigenvalues to construct the transformation matrix T. i.e.,

$$\mathbf{T} = \begin{bmatrix} \mathbf{t}_1 \\ \vdots \\ \mathbf{t}_n \end{bmatrix} \tag{2.17}$$

Where T is a $n \times N$ matrix and n can be chosen from 1 to N.

Then the transformation F can be expressed as:

$$\mathbf{F} = \mathbf{T}\mathbf{b}' \tag{2.18}$$

and

$$\mathbf{b}' = \mathbf{T}'\mathbf{F} \tag{2.19}$$

The KL transform can be used in transform image coding because of its energy compaction property. That means that it is possible to code only a fraction of the transform coefficients without seriously affecting the image. The KL transform is also useful in pattern recognition because the coefficients corresponding to the large eigenvalues can be selected as features for classification and recognition.

2.4 Object Isolation

In object isolation, there are several algorithms which can be used to separate the images of individual objects from the rest of the scene [2]. Those algorithms can be thought of from two different philosophical perspectives. One is the region approach, which assigns pixels to particular objects or regions; the other is the boundary approach, which attempts to locate the boundary that exists between the regions. Thresholding is a particularly useful region approach technique for scenes containing solid objects within a contrasting background. When using a threshold rule for image segmentation, assuming a "dard" background, all pixels with a gray level at or above the threshold are assigned to the object. All pixels with a gray level below the threshold are assigned to outside of the object. To reduce the background noise, object connectivity, which will connect all the pixels on the border of an object, is done by using an eight point border following method.

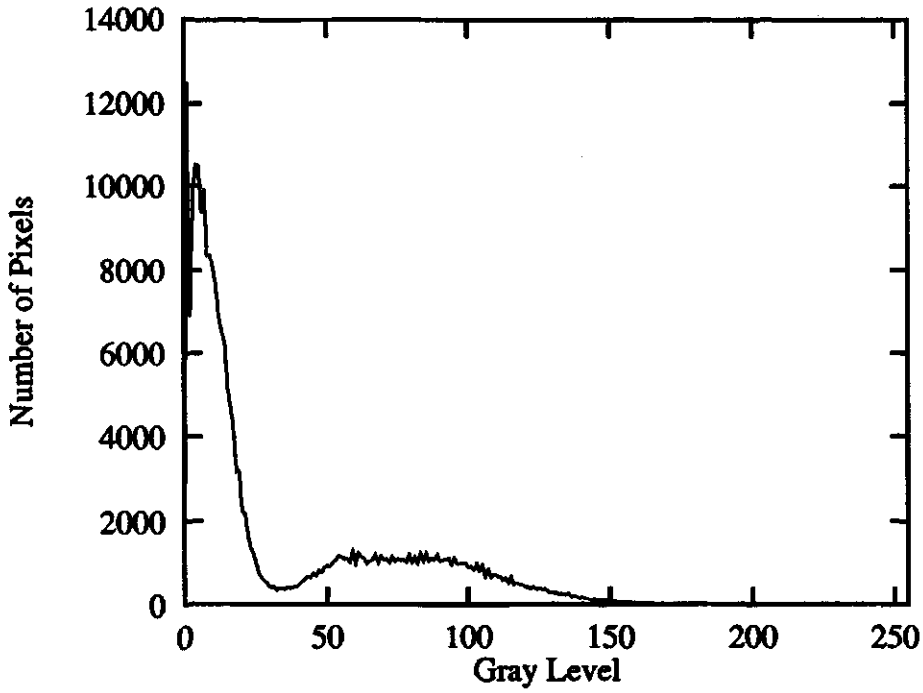


Figure 2.5: The bimodal histogram

Histogram Techniques

The histogram of an image, denoted by $H(I)$, represents the number of pixels which have a specific intensity, I. An image containing an object on a contrasting background has a bimodal gray level histogram as shown in Fig 2.5. The minimum value between the peaks is used to establish the gray level threshold, T. The total area of all objects defined by a gray level greater than a threshold T is given by

$$A = \int_T^\infty H(I)\, dI \tag{2.20}$$

if the area A only changes slightly when the threshold is changed from T to $T + \Delta T$, the selection of T is adequate. Thus placing the threshold at the minimum in the histogram minimizes the error in object isolation.

Connectivity

After calculating the threshold of an image, the original image is converted into a binary image. Pixels with values above the threshold are assigned to 1's while pixels with values below the threshold are assigned to 0's. The edges of objects can be found by connecting the border points which surround an object or a set of objects as an identifiable single object. A border following algorithm can be applied to find the connectivity of objects. This algorithm has an advantage of fewer computational steps compared to other algorithms, such as the edge-following algorithm [19]. The algorithm is briefly summarized as follows. In the binary image, the object pixels are

1's and the background pixels are 0's. The border element is defined as a pixel which has the value of 1 and at least one zero neighborhood point. To find the border of an object, an initial border element has to be found first. For example, pixel d3 in Fig 2.6 can be selected as an initial border element. As a border element, d3 has at least one zero neighborhood point, such as d2. Therefore, the initial edge is between d3 and d2. The pixel d2 is defined as the starting search point to find the next border element. The neighborhood points are searched in counterclockwise order from d2 to c2, c3, c4, d4, e4, e3, and e2. The first none-zero pixel reached during searching, such as c4 in Fig 2.6, is the new border element. The zero pixel before reaching the new border element is the new starting search pixel. The border points are traced by repeating the above algorithm until the initial border element is reached again.

In a special case, after an initial border element, if all neighborhood pixels are 0's, the initial border element is a sole pixel(e.g. b2).

2.4.1 Neural Networks

Neural networks provide an effective approach for a number of applications to solve problems involving patterns, such as pattern mapping, pattern completion and pattern classification.

There are some advantages of using neural networks as a classifier. Based on the learning algorithm, a neural network maps the input space to the classified output

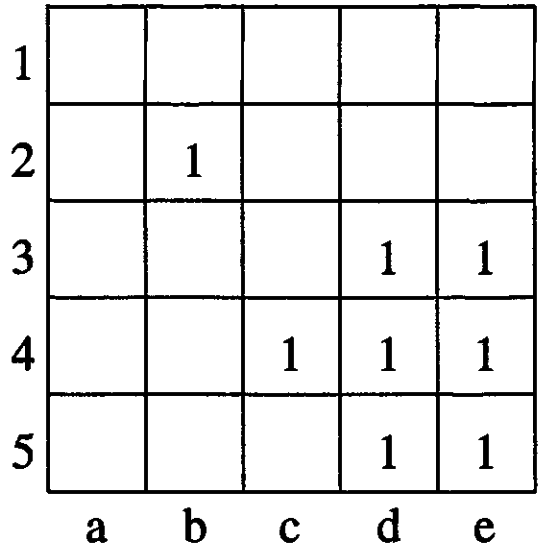


Figure 2.6: The border tracing

space. It can be used to do very complicated mapping jobs without a complete understanding of the underlying fundamentals of the process under study. Since neural networks are fundamentally parallel processing, it is possible to implement them in hardware, which could make significant increases in processing speed. A neural network can provide an "on line" environment for pattern classification as opposed to "off line" techniques that rely on a *priori* knowledge and statistical properties of the input data set, if the neural network is subjected to ongoing training.

The neuron

The basic processing unit for a neural network is called a neuron. A neuron can be viewed as a mathematical processor which maps the neural input vector, $\mathbf{x}$, into

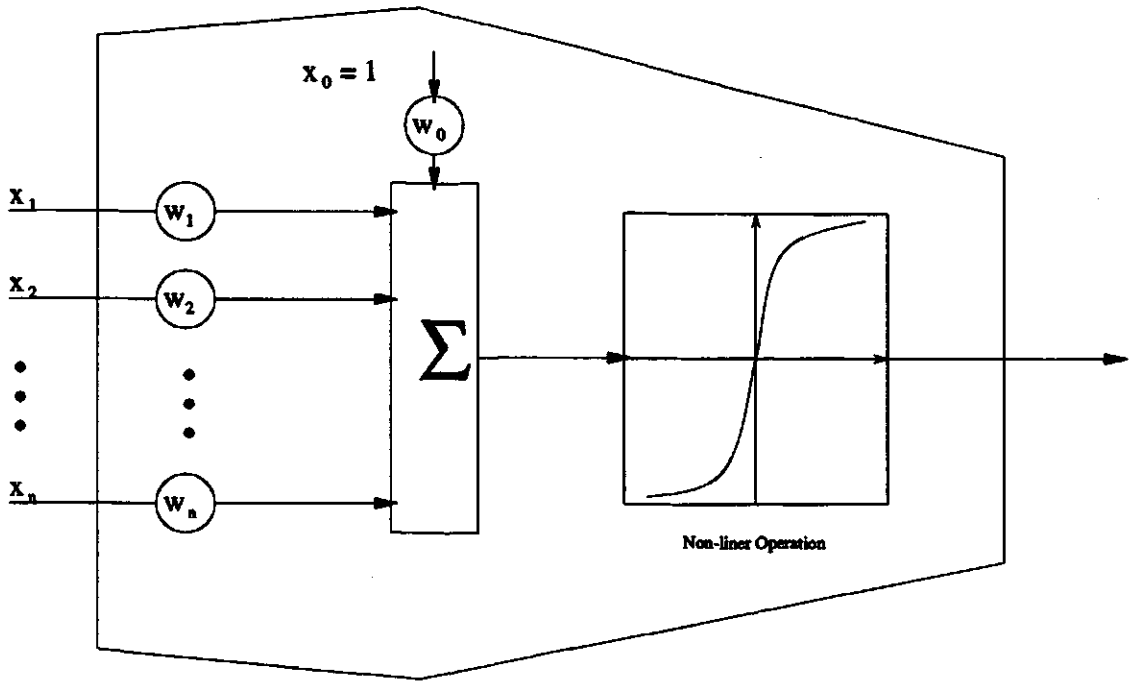


Figure 2.7: The mathematic model of a neuron

a single neural output, y, as shown in Fig 2.7. On the left are the multiple inputs x to the neuron. Each input has an associated connection strength, given in a weight vector w, which is determined from past experience (knowledge or memory). The neuron multiplies the inputs with their respective weights, adds them and applies a nonlinear threshold function $f()$ to the sum, in order to compute its output. The function $f()$ is also called the neuron activation function. The calculated result y is sent as the output of the neuron. The weight set $\{w_i\}$ is updated through a learning process.

The mathematical expression of a neuron is written as:

$$y = f\left(\sum_{i=1}^{n} w_i x_i - w_0\right) \tag{2.21}$$

where $f()$ is the nonlinear function and w_0 is the threshold value.

The nonlinear activation function $f()$ could have many shapes. All variables are real numbers and there are no bounds set for the neural network inputs. The activation functions are *sigmoid* (S-shaped) functions, such as a *logistic* function or a *hyperbolic tangent* function. The advantage of sigmoid functions is that they are differentiable, which is required by some training algorithms, such as the backpropagation algorithm. The *logistic* function can be expressed as:

$$f(x) = \frac{1}{1 + e^{-\alpha x}}, \tag{2.22}$$

and the *adjusted hyperbolic tangent* function can be expressed as:

$$f(x) = \frac{e^{\alpha x} - e^{-\alpha x}}{e^{\alpha x} + e^{-\alpha x}}. \tag{2.23}$$

where α is a constant controlling the slope of the curve. When α equals 1, it becomes the regular *hyperbolic tangent* function.

Multi-layer neural network

A multi-layer neural network consists of a set of neurons which are arranged into two or more layers: an input layer, an output layer, and at least one hidden layer. The input layer is usually not counted as a true layer since the input nodes are

not necessarily neurons. The signal flow from the input layer to the output layer is called *forward propagation*. The forward propagation step is initiated when an input pattern is presented to the network. Each node in the input layer directly takes a corresponding input value. Only the hidden and output neurons operate on the inputs according to Eq 2.21 and generate an output. The neurons in each layer take inputs from the outputs of neurons in the previous layer and pass the outputs to the neurons in the following layer. A three layer neural network which contains an input layer, two hidden layers and an output layer is shown as Fig 2.8. Nearly always, the same activation function is used for all neurons. The behaviour of the neural network is primarily determined by the weights which connect neurons in one layer to those in the next layer.

Learning algorithms and neural network training

There are many learning algorithms that can be used with neural networks. Backpropagation is one of the most widely used algorithms and is also one of the easiest algorithms to understand. It is used under supervised learning, which is meant to be a training directed by known patterns. The backpropagation algorithm is an iterative gradient algorithm designed to minimize the mean square error between the actual output of a multi-layer neural network and the desired output. It requires a continuous differentiable activation function. The principle of backpropagation is to adjust

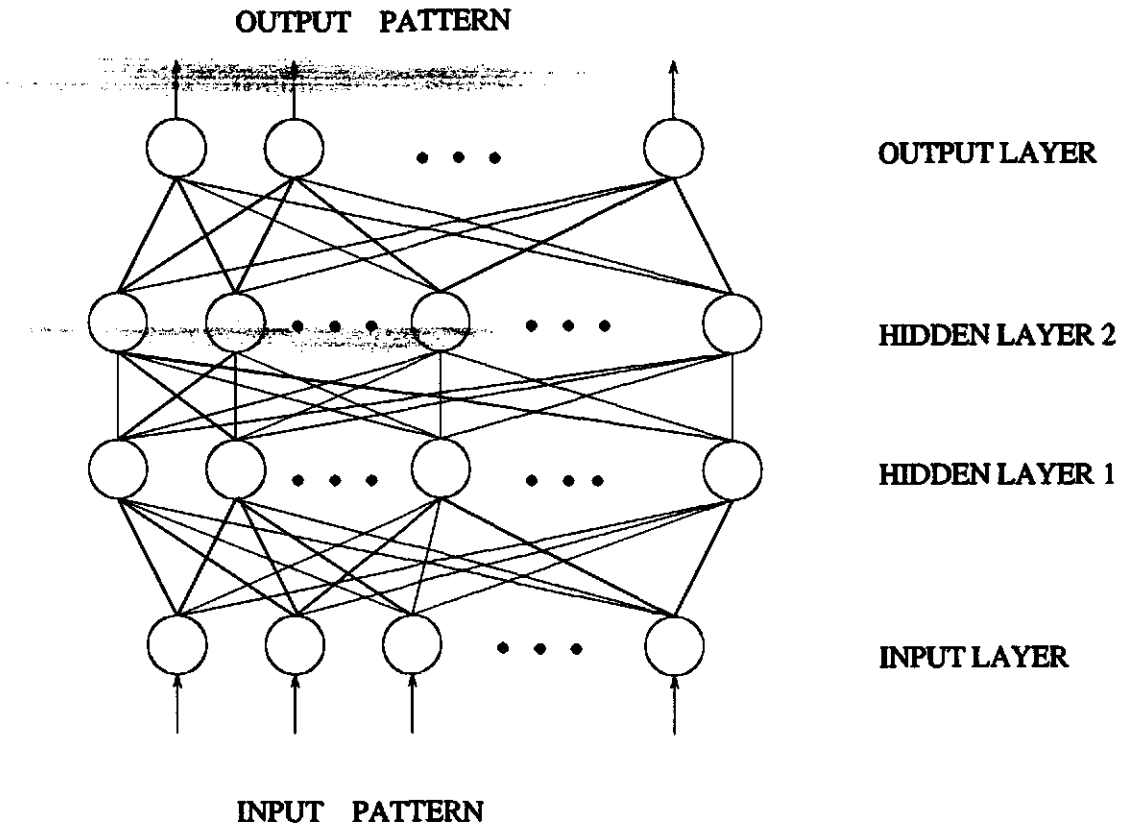


Figure 2.8: Three layers neural network

the weights in the direction opposite to the instantaneous error gradient, where the error is defined as the difference between the actual output and the desired output. A more complete description of the backpropagation algorithm is given in Appendix A.

The purpose of backpropagation training is to reduce the total error at the output layer. The training process starts by initializing all weights to small non-zero values, usually generated randomly. To illustrate the training of a backpropagation neural network, the network is provided with input/output pair patterns, each input pattern corresponding to a desired output pattern. Fig 2.9 gives an example of a training pattern set. The input patterns are shown along with their corresponding desired output patterns. Each input pattern consists of four numbers which represent four different features of one class of objects. In Fig 2.9, three classes of objects are represented as (0.25, 0.75, 0.75, 0.25), (0.25, 0.25, 0.75, 0.25), and (0.25, 0.25, 0.25, 0.75). The corresponding output patterns are defined as 1, 0.5, and 0 respectively, and the neural network needs to be trained to map the input patterns to the corresponding output patterns. A backpropagation training algorithm involves a forward-propagation followed by a backpropagation process; both of these operations are performed for each pair of patterns during training. The training process begins with forward-propagation by providing an input pattern and getting the calculated output pattern. By calculating the instantaneous error gradient between the actual

output and the desired output, the network adjusts its weights according to the back-propagation algorithm. The different patterns in the training set are presented to the neural network repeatedly, and the training iteration is completed when all patterns in the training set have been presented. At each training iteration, each input/output pattern pair is presented once. Multiple iterations must be executed in order to further reduce the output error. Usually, successful training of a backpropagation neural network might require hundreds or thousands of training iterations.

Usually, the root-mean-square (RMS) error is used as a quantitative measure of learning. It is computed by the equation

$$e = \sqrt{\frac{\sum_i \sum_j (d_{ij} - y_{ij})^2}{n_t n_o}} \tag{2.24}$$

where n_t is the number of patterns in the training set, n_o is the number of neurons in the output layer, $\sum_i$ is the summation of all patterns in the training set, $\sum_j$ is the summation of all output neurons, d_{ij} is the desired value for output neuron j after presentation of pattern i, and y_{ij} is the actual value for output neuron j after presentation of pattern i

As a neural network learns, the RMS error decreases. Generally, when the RMS error decreases below 0.1 for inputs normalized to unity, the neural network has learned its training set.

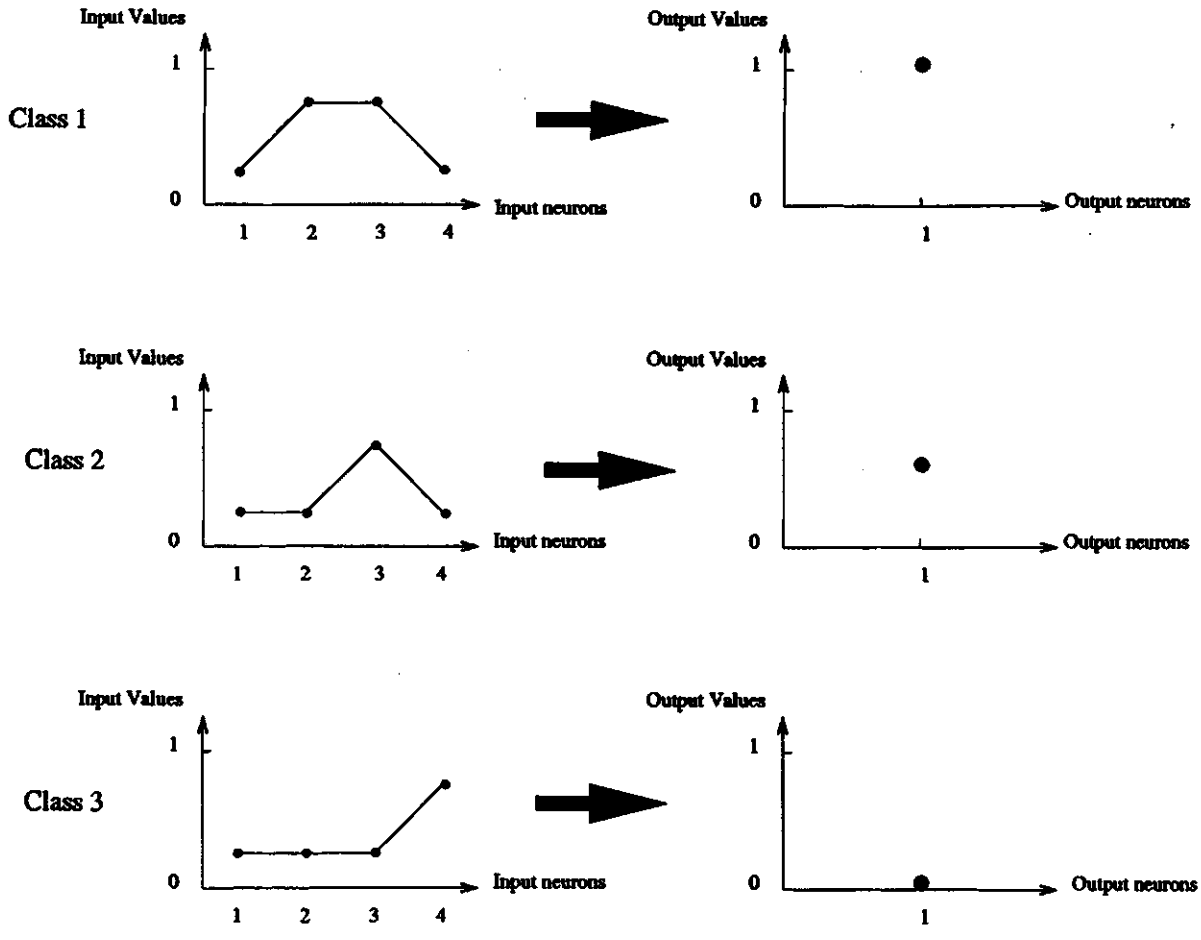


Figure 2.9: Detailed description for training set

Network Architectures

Generally, it is difficult to configure an effective network architecture. Difficulty arises when one tries to determine how many hidden layers and how many hidden neurons are appropriate for a specified problem. These numbers must be determined by repeated experiments since there is no known analytical method to determine their values [8], but there are some rules of thumb for choosing an appropriate network. It has been verified theoretically that there is no reason ever to use more than two hidden layers [8]. For most practical problems, it is recommended that one hidden layer be the first choice. For a neural network which has many inputs and few outputs, the number of neurons forms a pyramid shape, with the number decreasing from input toward the output. For a neural network which has few inputs and few outputs, and the problem is complicated, more hidden neurons may be required.

2.5 Image Statistics and Huffman Coding

2.5.1 Image Statistics and Entropy

Efficient image coding can be accomplished by first determining the structure of the image data and then developing encoding algorithms that are efficient for that data structure. One manifestation of an image is in the image statistics. Statistical redundancy of an image can be considered in two different ways. The redundancy

is significant when the autocorrelation function of the image is highly concentrated. The redundancy is also high when the probability density function of the image is not uniform. These two types of redundancy may be removed by using proper image coding algorithms. The amount and the nature of redundancy can be defined in terms of various statistical parameters characterizing the image.

Image Statistics

A digital image with a vertical size of N_1 pixels and a horizontal size of N_2 pixels is represented by an $N_1 \times N_2$ element matrix

$$\mathbf{F} = \{F(n_1, n_2)\}, \tag{2.25}$$

where $1 \leq n_1 \leq N_1$, and $1 \leq n_2 \leq N_2$. For an 8-bit gray level digital image, the possible values of $F(n_1, n_2)$ are from 0 to 255. The probability of occurrence of the event $F(n_1, n_2) = i$ is defined as:

$$P_i = P\{F(n_1, n_2) = i\} \tag{2.26}$$

where i is the pixel value from 0 to 255.

If the correlation between the pixels is not considered, P_i is estimated by first-order histogram as:

$$P_i \approx \frac{N_i}{N} \tag{2.27}$$

where N_i is the number of pixels whose value is i, and N is the total number of pixels

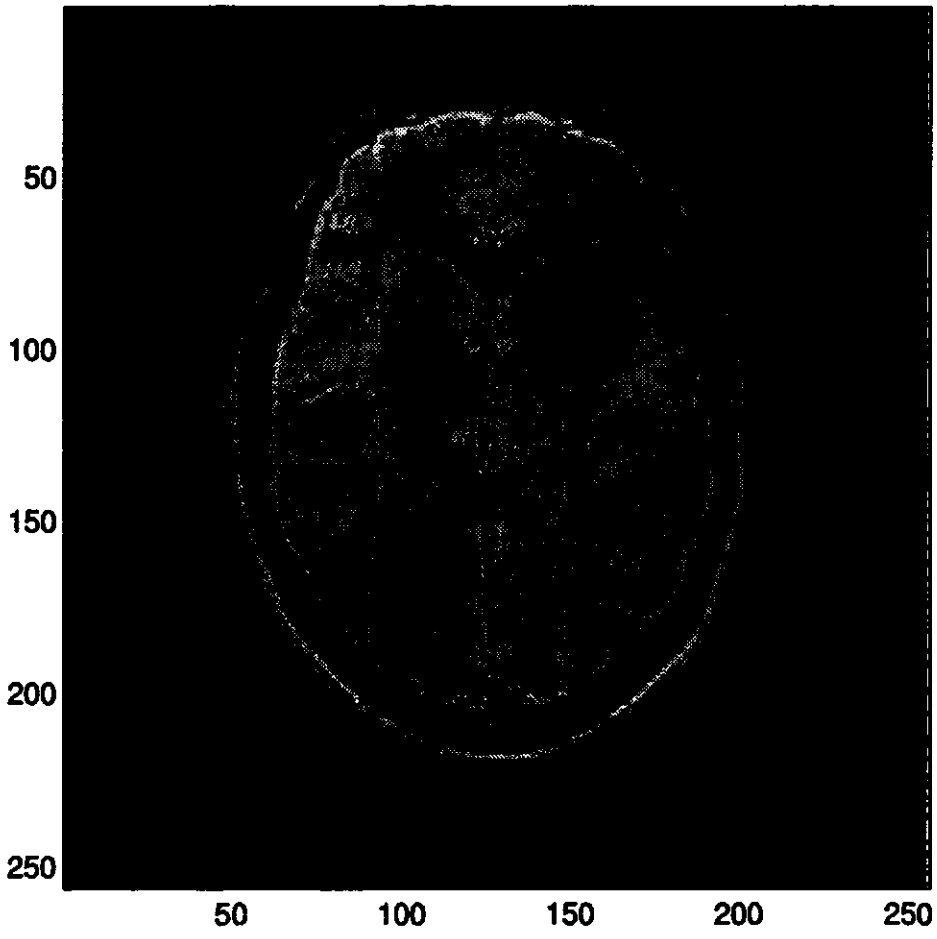


Figure 2.10: The MR image of a human head

of the image. Fig 2.11 illustrates a probability density function of an MRI image of the human head shown in Fig 2.10.

Adjacent pixels in a digital image are highly correlated. *Joint probability* takes adjacent pixels into account. It is defined as the probability for the first pixel (n_1, n_2) to take a value i and the adjacent pixel $(n_1 \pm 1, n_2)$ or $(n_1 n_2 \pm 1)$ to take a value j,

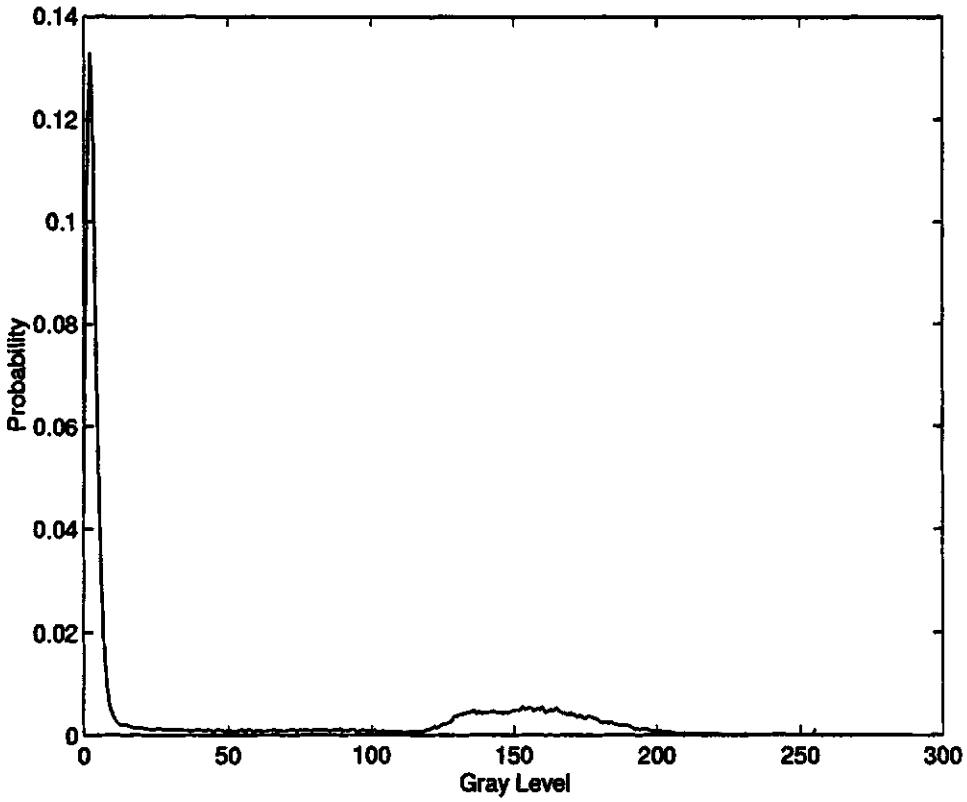


Figure 2.11: The probability distribution function of a human head image

i.e.

$$P_{i,j} = P\{F(n_1, n_2) = i,\ F(n_1 \pm 1, n_2) = j\} \tag{2.28}$$

or

$$P_{i,j} = P\{F(n_1, n_2) = i,\ F(n_1, n_2 \pm 1) = j\} \tag{2.29}$$

$P_{i,j}$ is estimated from the joint histogram as:

$$P_{i,j} \approx \frac{N_{i,j}}{N}$$

where $N_{i,j}$ is the number of pixel pairs with values i and j, respectively, and N is the total number of pixel pairs of the image.

Information entropy

Subjectively, it is obvious that some images contain more "details" or perhaps they yield more "data" than others. Details, data, and similar terms are used to describe the state of an image. Those terms are vague and qualitative. To represent image information quantitatively, the average information of the image is estimated by its first-order entropy H, defined as:

$$H = -\sum_{i=1}^{I} P_i \log_2 P_i \tag{2.30}$$

where P_i is the probability associated with the occurrence of a value i, and I is the total number of categories. From the probability distribution function, the information entropy of an image can be calculated.

By considering two adjacent pixels together, a more accurate calculation of image information content could be made. The joint entropy takes the correlation of two adjacent pixels into account. It is based on the joint probability $P_{i,j}$. The probability for a pixel x to take a value i is given by:

$$P_i = \frac{N_i}{N} , \tag{2.31}$$

the probability for a pixel y to take a value j is given by:

$$P_j = \frac{N_j}{N} , \ and \tag{2.32}$$

the joint probability for the first pixel x to take a value i and the pixel y to take j is given by:

$$P_{ij} = \frac{N_{ij}}{N} . \tag{2.33}$$

The entropies based on the joint probability and the conditional probabilities are then obtained as follows:

$$H(x,y) = -\sum_{i,j} P_{ij} \log_2 P_{ij} , \tag{2.34}$$

$$H(x|y) = -\sum_{i,j} P_{ij} \log_2 \frac{P_{ij}}{P_j} , \ and \tag{2.35}$$

$$H(y|x) = -\sum_{i,j} P_{ij} \log_2 \frac{P_{ij}}{P_i} . \tag{2.36}$$

The entropy $H(x,y)$ is divided by 2 to translate it into units of the entropy per pixel.

It is well known that [16]

$$H(x,y) = H(x) + H(y|x) = H(y) + H(x|y) . \tag{2.37}$$

Theoretically, it can be shown that [16]

$$H(y|x) - H(y) \leq 0 \,, \tag{2.38}$$

which means that the entropy calculated for the second pixel y given the occurrence of the first pixel x is never greater than the entropy calculated from y alone. So,

$$H(x,y) \leq H(x) + H(y) \,, \; and \tag{2.39}$$

$$H(x,y) \leq 2H(x) \tag{2.40}$$

This means the entropy calculated based on the joint probability is always equal to or less than the entropy based on the first-order probability distribution. The equality holds only when the pixels of an image are completely independent.

2.5.2 Source Coding

Source coding is used to code the image intensity itself or the result of simple modification to the image intensity, such as a logarithmic intensity. To reduce the statistical redundancy of an image, pixel differences are often used. In the case of 2D images, there are four (up, down, left and right) or eight (if diagonal differences are considered) possible directions in which a pixel difference can be calculated. For viewing an image and progressively improving its quality as more data become available through transmission, there are some algorithms for constructing an image from differences in an hierarchical manner. More detailed descriptions about progressive

transmission are given in Chapter four. One major advantage of source coding is its simplicity.

2.5.3 Codeword Assignment and Huffman Coding

The objective of codeword assignment is to assign a specific codeword(a string of 0s and 1s) to each symbol of the source alphabet. The codewords must be uniquely and instantaneously decodable. That means each symbol of the source alphabet must be assigned a different codeword, and in addition, since the symbols are transmitted in sequence, the codewords have to be designed so that they can be identified when received sequentially. In order to make the codewords instantaneously decodable, no codeword in the codes can be a prefix of any other codewords [17]. In general, the prefix condition requires that for a given codeword of length k, there is no other codeword of length $l < k$ which is identical to the first l binary digits of this codeword.

The codeword assignment can be divided into *uniform length codewords*, in which each symbol is assigned a codeword of the same length, and *variable length codewords*, in which codeword lengths are changed depending on the probability of occurrence of the symbols. Variable length coding, Huffman coding in particular, is in general optimal in terms of the required bit rate or in the compression gain.

Huffman coding is described by the following example: given a source with alphabet $A = \{a_1, a_2, \cdots, a_8\}$, the probability corresponding to the symbols a_1, a_2, $\cdots$, a_8

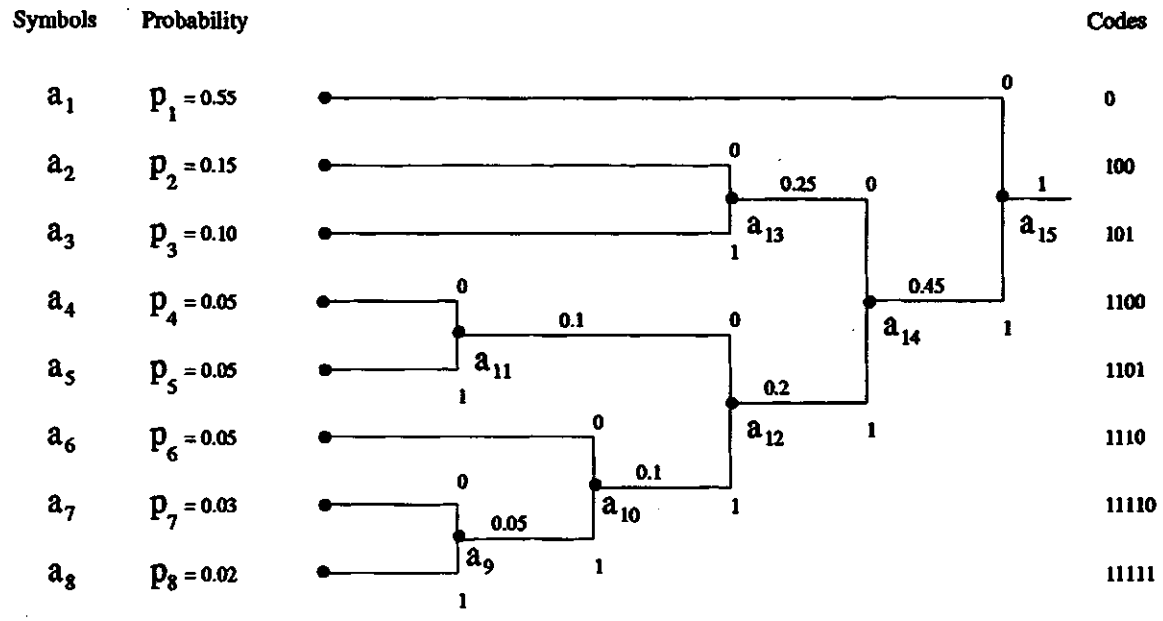


Figure 2.12: An example of Huffman coding

is p_1, p_2, $\cdots$, p_8, respectively. The symbols are listed in decreasing order of their probability, as shown in Fig 2.12. Each black dot in Fig 2.12 is defined as a node. In the first step of Huffman coding, the last two symbols, a_7 and a_8, which have the lowest probabilities are combined to form a new node with the combined probability. 0 is assigned to one of the two branches leading into this node and 1 is assigned to the other branch. Now we consider a_7 and a_8 as a new symbol a_9 with probability 0.05. Among the seven symbols, a_1, a_2, a_3, a_4, a_5, a_6 and a_9, the two symbols (a_9 and a_6) with the lowest probabilities are combined as one symbol again (a_{10}), and 0 is assigned to one branch and 1 is assigned to the other. This procedure is repeated until one symbol with the probability 1 is left. The codewords are determined by

tracing the branches from the last node to the symbols and combining the 0's and 1's. The codewords for each of the symbols are also shown in Fig 2.12.

In this example, the entropy of the source is calculated as $-\sum p_i \log_2 p_i$, which is 2.13(bits/symbol). The average bit rate of Huffman coding is calculated as:

$$\frac{\sum(\ probability\ of\ each\ symbol \times corresponding\ Huffman\ code\ length\ )}{number\ of\ sysbols}, \quad (2.41)$$

which is 2.15(bits/symbol). If uniform length coding is applied to the above example, one possible set of codewords is shown in the Fig 2.13. The bit rate using uniform length codewords is 3 bits/symbol. From this example, we could see that Huffman coding is much better than uniform length coding when the symbols in the source have different probability. In general, Huffman coding is optimal in the sense that the average bit rate required to represent the source symbols is minimum.

2.6 Summary

This chapter has introduced some basic concepts and techniques in image analysis. An eight bit image is obtained by sampling an analogue image and scaling the intensity between 0 and 255. Colour images could be represented in different colour coordinate systems. The KL transform, which could be used as a reference for choosing the proper colour coordinate to represent the different kinds of objects, was introduced. The threshold image segmentation technique and eight-point border following connectivity

Symbols	Probability	Codes
a1	p1 = 0.55	000
a2	p2 = 0.15	001
a3	p3 = 0.10	010
a4	p4 = 0.05	011
a5	p5 = 0.05	100
a6	p6 = 0.05	101
a7	p7 = 0.03	110
a8	p8 = 0.02	111

Figure 2.13: Codewords of uniform length coding of above example

technique were introduced as suitable tools to isolate objects from the background in visual pattern recognition. The neural network was introduced as a mathematical model. The supervised learning algorithm used to adjust the weights in the multi-layer network called backpropagation was outlined in Appendix A. The general method of supervised multi-layer neural network training was described.

Some basic concepts and techniques in image coding were introduced in this chapter. To obtain efficient image coding, image statistics were studied. The Huffman coding technique, which is optimal in terms of the bit rate, was also described.

Chapter 3

Seed Classification

3.1 Introduction

Image analysis has been widely applied in many areas; visual pattern recognition is one example. In the visual pattern recognition approach, it is assumed that each image may contain one or more objects and that each object belongs to one of several predetermined types or classes. The attributes of these objects are measured by applying image analysis methods. The objective of the work presented in this chapter is to classify different barley seeds based on their colour features. This chapter presents a colour discrimination system for barley seed variety classification, including the classification procedures and experimental results.

The discrimination system consists of three steps: image capturing, feature extraction, and pattern recognition. The hardware portion of the image capturing system consists of a Macintosh IIfx computer with a Data Translation QuickCapture image acquisition board, a WILD M8 zoom stereomicroscope, and a Javelin colour video camera.

An enhanced version of the NIH Image software package, *Colour Image*, was used as the image capturing software. The barley seed feature extraction step was done in two stages consisting of barley seed isolation and feature selection. The pattern recognition uses a neural network as the classifier. Training the neural network is performed prior to the actual classification. The classification accuracy is monitored to evaluate how well the neural network has been trained.

The procedure flow chart for barley seed classification is shown in Fig 3.1. There were four kinds of barley seed samples: Tankard, Breedn, Creme and CDC Dolly. Images of samples of these four different barley varieties were captured to construct training data sets to be used in training a multi-layer neural network and test data sets to be used in evaluating the classification accuracy of this system. Feature extraction and pattern recognition were performed on a Sun work station.

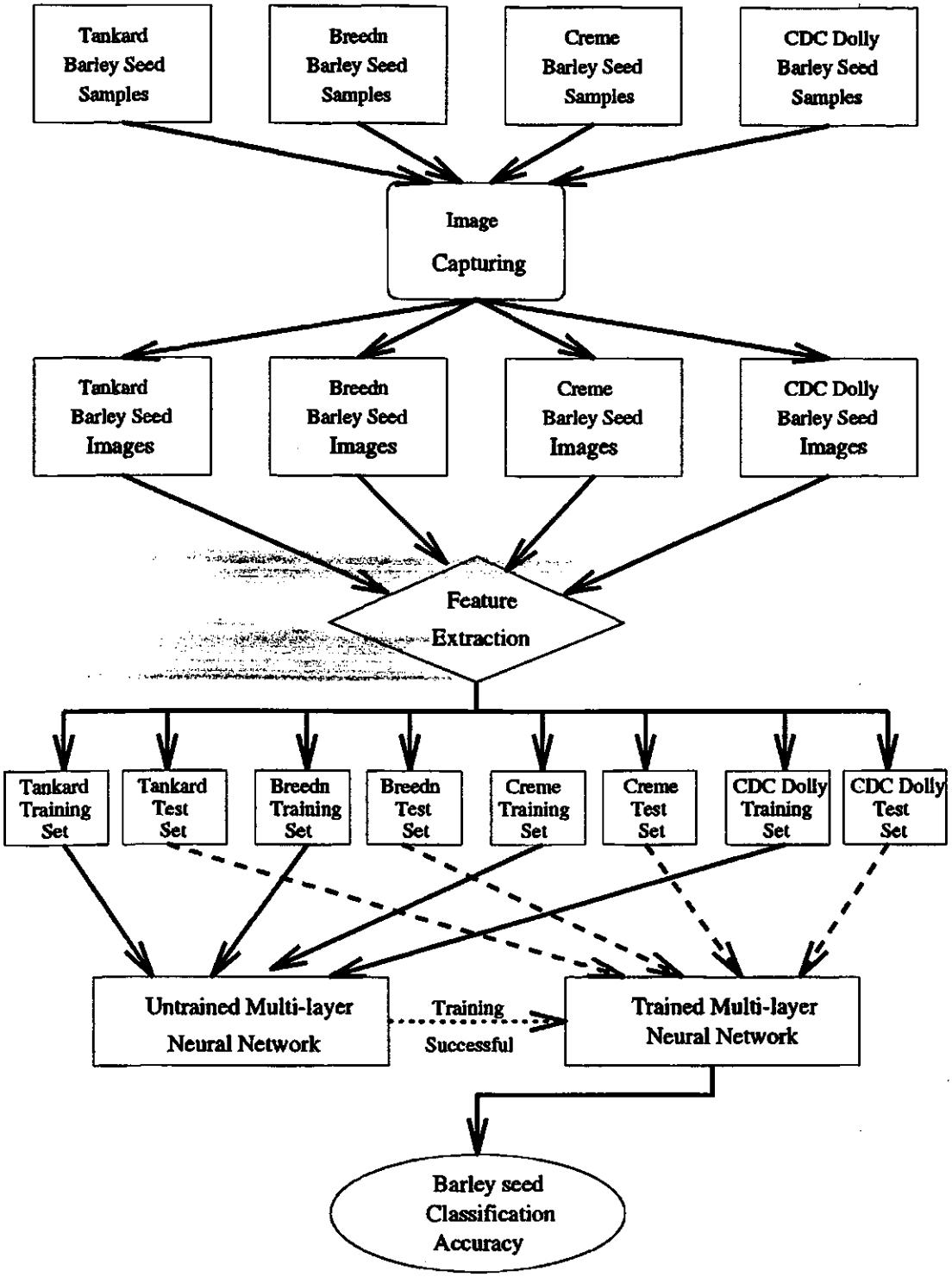


Figure 3.1: The procedure flow of the classification of four kinds of barley seeds

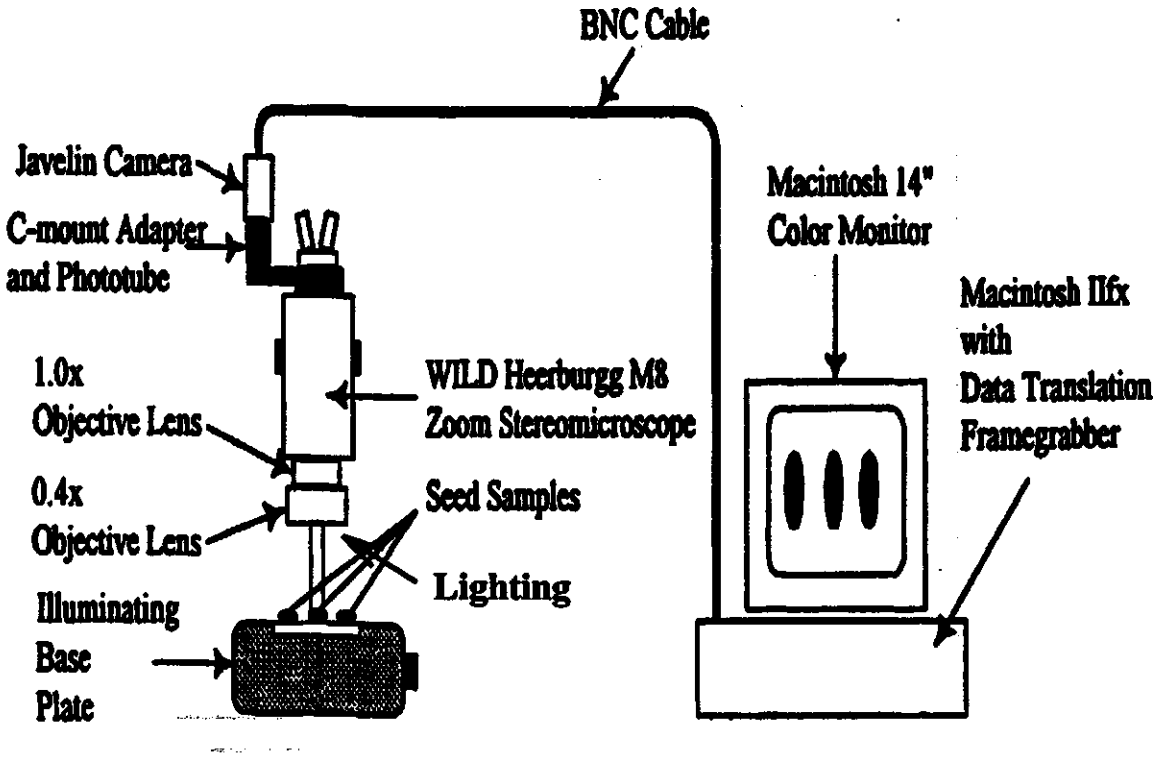


Figure 3.2: Machine vision setup for seed analysis (from[16])

3.2 Machine Vision System

The machine vision system used to obtain seed images has a configuration as shown in Fig 3.2 [18]. It consists of five parts: a video camera, a framegrabber, a computer, a lighting assembly, and a microscope. The seeds were placed on a base plate, the light source was set above the seed samples, and the images were captured by a video camera connected to an image capture board (framegrabber).

3.2.1 Hardware and Software

Platform and Framegrabber

A Macintosh IIfx computer was used as the image acquisition platform. It had a 40MHz CPU, a 160MB hard disk, and 8MB of memory. A 14" Macintosh colour monitor and a 21" Macintosh black and white monitor were used to display captured images. A QuickCapture framegrabber board was installed on one of the NuBus slots of the Macintosh computer.

The framegrabber receives an NTSC analogue video signal and digitizes a frame of video image by A/D conversion to produce digital images. The framegrabber captures a series of video frames at a rate of 30 frames per second. In gray scale image acquisition, analogue signals are converted into an 8-bit digital format which provides 256 gray levels. The gray scale values from 0 to 255 represent different degrees of grayness. The gray scale value "0" corresponds to black while the value of "255" corresponds to white. In colour image acquisition, colour images was obtained by using an RGB camera with a four channel multiplexor. The size of grabbed images was 768 pixels (horizontal) times 512 pixels (vertical).

Image Acquisition Software

NIH Image is a public domain image processing and analysis software package for Macintosh computers. It provides functions necessary for capturing and displaying

images obtained from the framegrabber. It supports the Data Translation QuickCapture card, and the Scion Image Capture 2 card for digitizing images. The image acquisition software used in this project is called *ColourImage*, an enhancement of version 1.29 of *NIH Image*. *ColourImage* is designed to acquire images from RGB video sources. Three grayscale images representing each of R, G, and B components can produce one true colour image.

A typical set of captured R, G, and B images of barley seeds is shown in Fig 3.3. The size of the images is 550 pixels (horizontal) times 440 pixels (vertical), which only covers the interested area.

Camera, Microscope and Lighting

The camera used is a Javelin JE3462RGB video camera. A WILD M8 Zoom Stereo-Microscope was used to replace the standard lens of the camera. The microscope provides stepless zoom magnification from 2.4 to 50. There are two choices of objective lenses, one is 1.0x and the other is 0.4x. The 0.4x lens was used to increase the field of view so that a picture can hold 10 barley seeds at one time without touching each other.

Lighting is an important factor in image acquisition. The lighting configurations and evaluation of these configurations are described by Hehn [4]. In this project, front lighting (the light source on the same side of the object as the camera) was used since

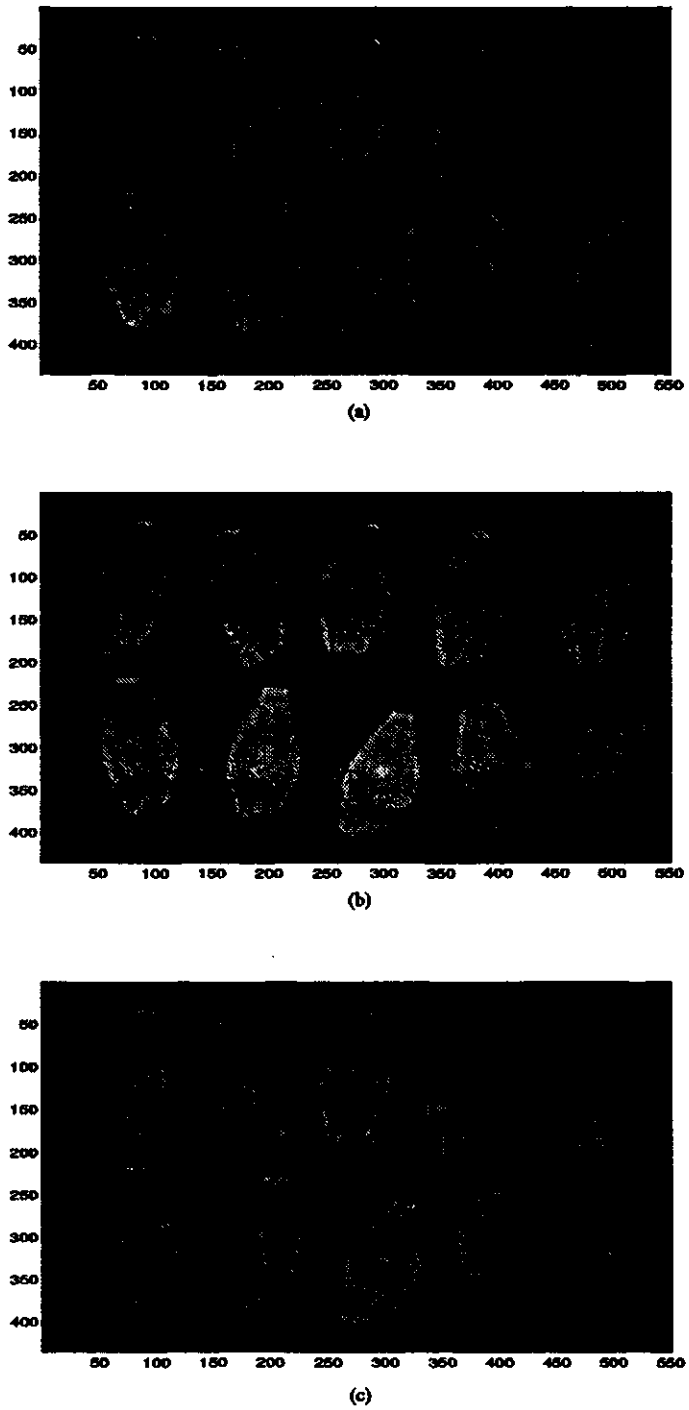


Figure 3.3: R, G, B images of 10 Breedn barley seeds where (a) is the image for the red component, (b) is the image for the green component, and (c) is the image for the blue component

the colour features of the reflectance is of primary interest. The seeds were detected and features were measured based on the reflected light. A fiber optic light ring (Volpi-Intralux 5000) was used in this project as the front lighting source. The fiber optic light ring was mounted on the front of the microscope to provide an incident light source when driven by a halogen lamp. The light intensity can be controlled manually. In our experiments, the light intensity was set to the maximum and kept no change during all the experiments.

3.2.2 Image Isolation and Feature Selection

Image Capturing

As shown in Fig 3.3, ten seeds at a time were placed on the base plate of the microscope. The front light source was used to illuminate the seed samples and the base plate. A black background was used to make the difference between the objects and background distinct in the images. The seeds were arranged in two rows of five seeds. For correct object isolation, seeds must not touch each other or the edges of the field. If seeds were placed touching each other or touching the edges of the field, the data of those seeds were discarded.

NIH ColourImage was used for capturing colour images. There are four channels out from the colour video camera. the red, green, and blue channels were digitized and a fourth channel was viewed on the live video monitor. The live video monitor

was used to monitor seeds placed in the field of view to make sure that all seeds were placed properly.

By using *NIH ColourImage*, R, G, and B components were captured individually. The corresponding R, G, and B images were saved in files r.dat, g.dat, and b.dat respectively.

Feature Extraction of Barley Seed

The objective of feature extraction is to obtain the distinguishable features which can be used to separate seeds into different classes. There are many attributes that could be used as features, such as area, ratio of length and width, etc. In this thesis, only the colour features of barley seeds are considered. Other features, such as shape, have been studied by other researchers [18]. Possible candidates of colour features are the mean values and the standard deviations of the R, G, and B values of each barley seed, and the R, G, and B peak intensity.

After an image is captured, the first step is to isolate the objects. Since the background was "dark" and the objects were "bright", the histogram technique was employed to isolate the objects. Fig 3.4 shows an example of histograms of R, G, and B components. By using the histogram technique described in Section 2.4, the threshold values of R, G, and B components were found to be 37, 43, and 34, respectively. Three binary images of R, G, and B components were generated based on those threshold

values. According to a basic multi-band image segmentation method [20], a combined binary image was produced by applying the AND operation on those three binary images, i.e. a pixel in the combined image is defined as a part of the object if its R, G, and B values are all greater than their thresholds, respectively. The object's boundaries were obtained from the combined image using the connectivity method described in Section 2.4. An image with object boundaries of Breedn seeds is shown in Fig 3.5. In the object extraction program, the mean values of R, G, and B components were also calculated.

Tankard, Breedn, Creme, and CDC Dolly seeds were measured using the system described above. Each seed variety was measured separately during the image capturing and feature extraction phase of the experiment. For each image, samples of ten seed samples were placed on the illuminated base plate of the stereomicroscope manually. The R, G, and B components were captured individually and sent to a Sun Sparc work station. Boundaries of seeds were obtained by using the feature extraction program.

As described in Chapter 2, the objective of feature extraction is a transformation to a pattern space from a measurement space (R, G, and B). The measurement space could be used as a feature space directly, but for best classification results, usually a transformation from the measurement space to a pattern space is desirable [9].

According to colour theory, there are several different colour spaces, such as RGB,

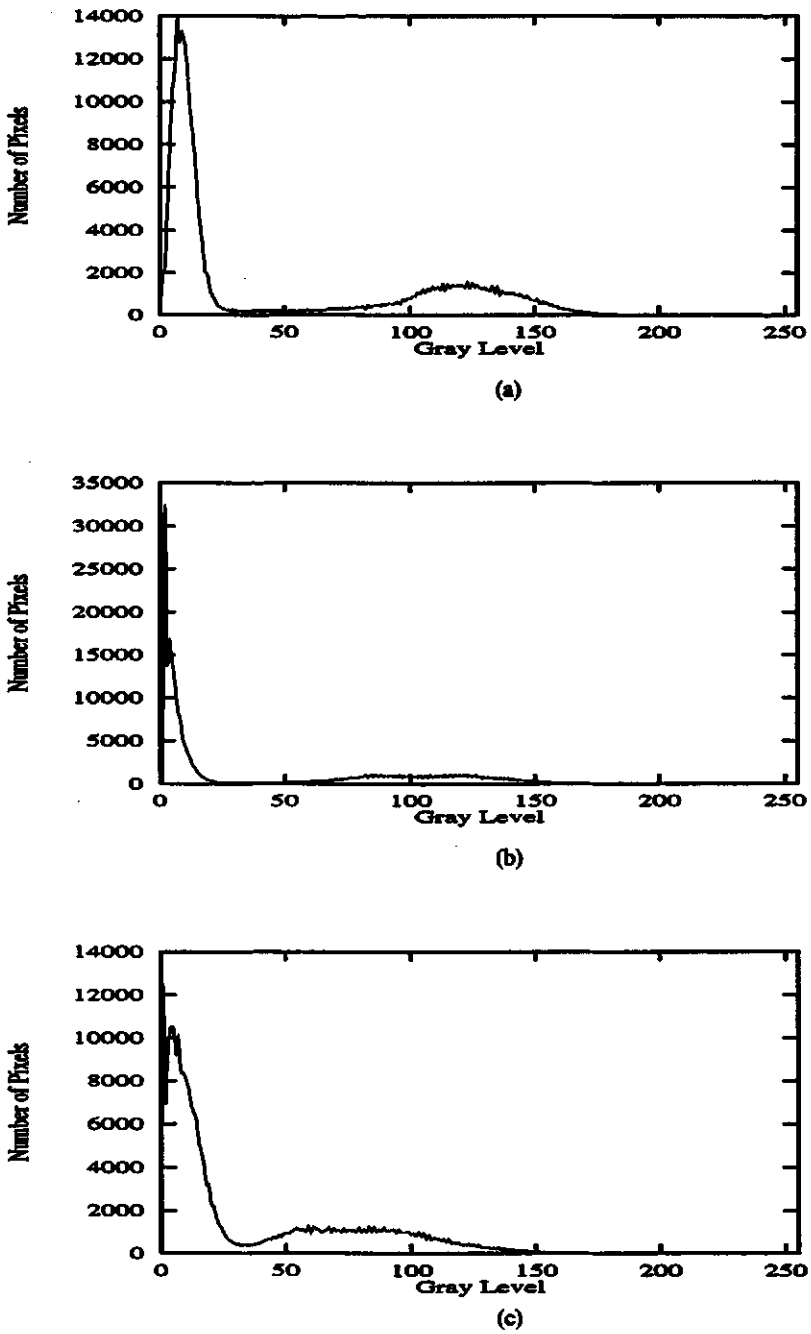


Figure 3.4: The histograms of R, G, B images of barley seeds, where (a) the histogram of red component (b) the histogram of green component (c) the histogram of blue component

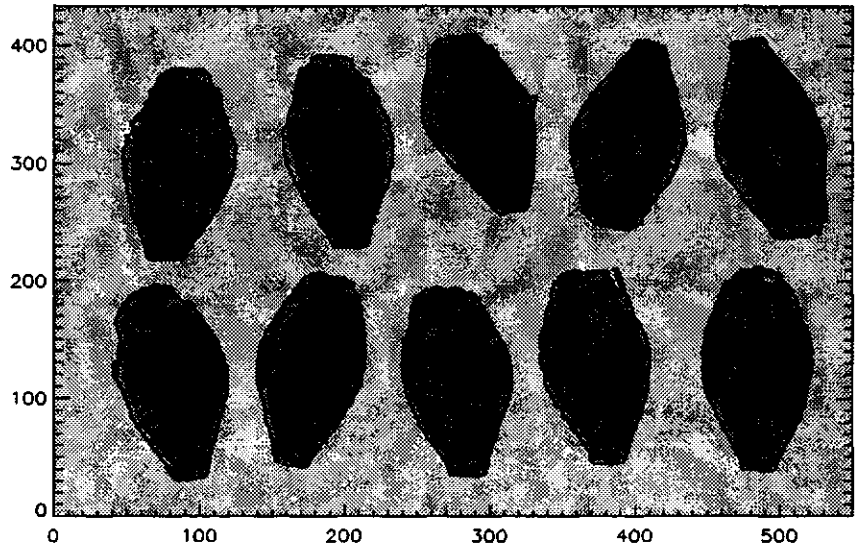


Figure 3.5: The image with object borders of Breedn seeds

HSI, Lab, etc. Among those colour spaces, HSI is more closely related to human perception, while Lab is a colour space which is uniform in the spacing of colours as related to visual difference. Therefore transformations from RGB measurement space to HSI space and Lab space were made. Using the same seed samples, different colour spaces were compared. From the number of experiments performed, it was found that the highest classification accuracy was obtained when the HSI space was used.

The reason why the HSI space is the best pattern space is investigated by applying the KL transform to RGB, HSI, and Lab spaces individually. The obtained R, G, and B mean values of each seed samples were transformed into HSI and Lab colour space according to Equations 2.8, 2.9, 2.10, 2.11, 2.12, and 2.13. The total number of seed samples was 560. The basic transform vectors of RGB colour space, i.e. the

orthonormal eigenvectors of the covariance matrix of the RGB data, are calculated as follows:

$$V_{RGB} = \begin{bmatrix} 0.4955 & -0.7299 & 0.4709 \\ -0.7665 & -0.1123 & 0.6323 \\ 0.4086 & 0.6743 & 0.6151 \end{bmatrix}$$

The basic KL transform vectors of HSI colour space are:

$$V_{HSI} = \begin{bmatrix} 0.8374 & 0.5465 & 0.0021 \\ 0.5466 & -0.8374 & -0.0022 \\ -0.0006 & -0.0030 & 1.0000 \end{bmatrix}$$

The basic KL transform vectors of Lab colour space are:

$$V_{Lab} = \begin{bmatrix} 0.2394 & 0.4247 & -0.8731 \\ 0.9169 & 0.1969 & 0.3472 \\ -0.3194 & 0.8836 & 0.3423 \end{bmatrix}$$

If the sequence in the original domain is already uncorrelated, the basic transform vectors should be:

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

From the above basic KL transform vectors of RGB, HSI, and Lab space, the transform vectors of the HSI space are the closest vectors to the identity matrix I. Therefore HSI space is the closest space to the ideal orthogonal space.

3.3 Neural Network Training and Testing

A multi-layer neural network was used as a classifier in the pattern recognition phase. The neural network program was written in the C programming language on a Sun Sparc work station by Broten [1]. This program was modified to fit the requirement of the seed classification study. The object isolation and feature extraction programs were developed in the C programming language on the same platform.

The neural network was trained to learn the relationship between the feature vectors and barley seed varieties. Two data sets were prepared for the learning phase; one was the training data set which contained the data to train the neural network, and the other was the test data set used to test whether the neural network was trained successfully.

3.3.1 Neural Network Operation

The behaviour and structure of the neural network is controlled by some important parameters such as the number of input nodes of the network, the number of output nodes of the network, the learning rate, μ, the type of the neuron activation function, the number of input/output vector combinations in the training and testing data sets, the desired error convergence before training stops, and the maximum number of epochs to be executed.

The feature vectors obtained from the feature extraction program were used as

the neural network inputs. The initial weights of the neural network were determined by random numbers in the range -0.2 to 0.2 [1]. The numbers of neurons in the two hidden layers were adjustable. The training and testing data sets were organized in a format of input/output vectors, i.e. a desired output vector line follows after each input vector line; values in each input/output vector line were separated by a space. Before starting neural network training, three files: a training data file, a testing data file, and a neural network initialization file were prepared. The parameters defining the neural network operation were loaded from an initialization file before the neural network started training. After training the neural network, a file containing the sum of squared errors after each iteration during training was created. According to the total summed square errors and the root-mean-square(RMS) errors, the performance of the neural network was analyzed.

The neural network used in this thesis is a multi-layer neural network. The number of the hidden layers must be one of 0, 1 or 2. The maximum number of neurons in each hidden layer is 50. The neuron activation function can be chosen either as a sigmoid function,

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3.1}$$

or a modified sigmoid function, which means:

$$f(x) = \begin{cases} 1 & \text{if } f(x) > T \\ -1 & \text{if } f(x) < -T \end{cases} \tag{3.2}$$

where T is the threshold of the activation function, which is typically set to be 0.97. The modified sigmoid function forces the result to converge more quickly than the normal sigmoid function. The training algorithm used in the neural network is a backpropagation algorithm. According to the original program by Broten [1], the learning rate can be chosen between 0.005 and 0.3 and the minimum adjustment learning rate was 0.0001. The maximum allowable number of training epochs of the neural network was 10000.

3.3.2 Training Goal and Learning Measurement

As described in Chapter 2, the neural network weights were adjusted during the training process in order to reduce the error between the desired output and the actual output of the neural network. In order to monitor the training status of the network, a set of test vectors was presented to the neural network after each training epoch. The total sum of the square errors was then calculated as follows:

$$E_{Total} = \sum_{i=1}^{n_t} \sum_{j=1}^{n_o} (d_{ij} - y_{ij})^2 \tag{3.3}$$

Where, n_t is the total number of test vectors, n_o is the number of outputs of the neural network, the subscript i is the index number of the testing vectors, j is the

index of neural network output, d_{ij} is the desired output and y_{ij} is the actual output. By using E_{Total}, a training curve was plotted in a graph of the training epoch versus the sum of squared errors. A typical training curve is shown in Fig 3.6. This training curve is based on a neural network which had three inputs, four outputs, 140 training samples and 140 testing samples.

The training goal is specified according to the shape of a typical neural network training curve. As shown in Fig 3.6, the training curve has a particular shape. At the beginning, the sum of squared errors of training is large since the weights of the neural network have been assigned randomly. Because of the large sum of squared errors, the weights are adjusted rapidly reducing the error by a large amount during the first 200 epochs. This rapid drop of the sum of squared errors is shown in Fig 3.6 during the first 200 epochs. In a successful training, the sum of squared errors continuously decreases, but the rate of reduction is rather slow. When the network is close to being fully trained, only a small amount of reduction of the summed squared error is achieved, as shown in Fig 3.6 where the error curve becomes nearly flat after 1000 epochs. The training goal is achieved when the decreasing rate of the sum of squared error is small enough.

A parameter called the root-mean-square (RMS) error is usually used as a quantitative measure of learning. The RMS error represents the average deviation of the neural network actual output values from the desired values. The RMS error is

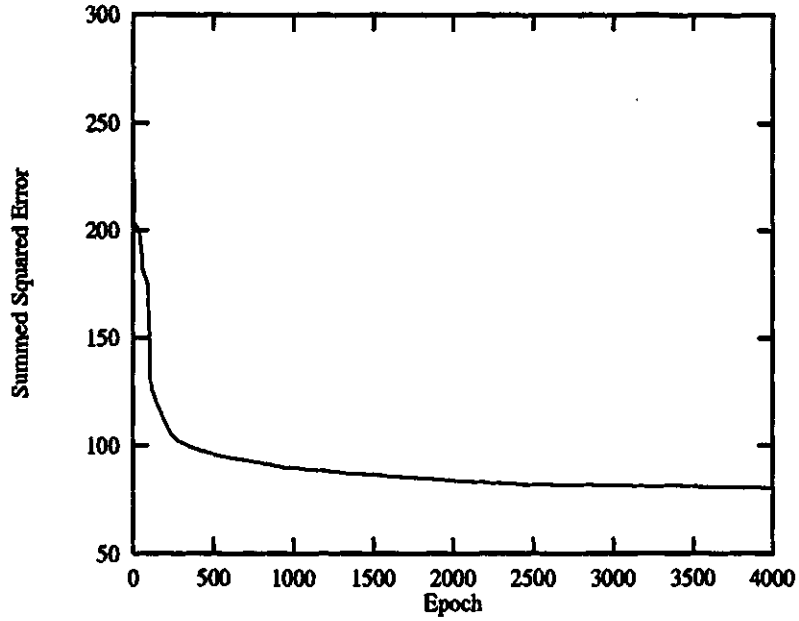


Figure 3.6: A typical training curve of the neural network

represented by the equation

$$e = \sqrt{\frac{\sum_{i=1}^{n_t} \sum_{j=1}^{n_o} (d_{ij} - y_{ij})^2}{n_t n_o}} = \sqrt{\frac{E_{Total}}{n_t n_o}} \tag{3.4}$$

Where, n_t is the number of test vectors, n_o is the number of neural network outputs, $\sum_{i=1}^{n_t}$ is a summation over all patterns in the test file, $\sum_{j=1}^{n_o}$ is a summation over all neural network outputs, d_{ij} is the desired output and y_{ij} is the actual output. E_{Total} is the sum of squared errors.

Table 3.1 gives a summary of the experimental results of the neural network training in terms of the RMS errors. Since the desired output value of the neurons in the output layer was either 0.1 or 0.9, these RMS errors were acceptable. The

Table 3.1: Summary of RMS errors

Colour Features	*# of Test Vectors*	*# of network outputs*	*RMS error*
HSI	560	4	0.182
Lab	560	4	0.183
RGB	560	4	0.194

barley variety recognition accuracy which corresponds to these results will be given in Section 3.4.

3.3.3 Barley Seed Discrimination and Recognition Accuracy

Fig 3.7 shows the input vectors for the neural network classifier, where the axes are the R, G, and B components of the barley seeds.

A successfully trained neural can obtain a correct output pattern for a given input pattern. When a neural network is used as a classifier, the desired output patterns are assigned to corresponding classes, respectively. In the current study, four barley varieties represented by four classes were classified. Two different output patterns were assigned and tested. In the first assignment of output patterns, only one neuron of the output layer was used. The four barley classes, which are Tankard, Breedn, Creme, and CDC Dolly, were assigned to the output values of 0.2, 0.4, 0.6, and 0.8 respectively, as shown in Fig 3.8. In the second assignment of output patterns, the

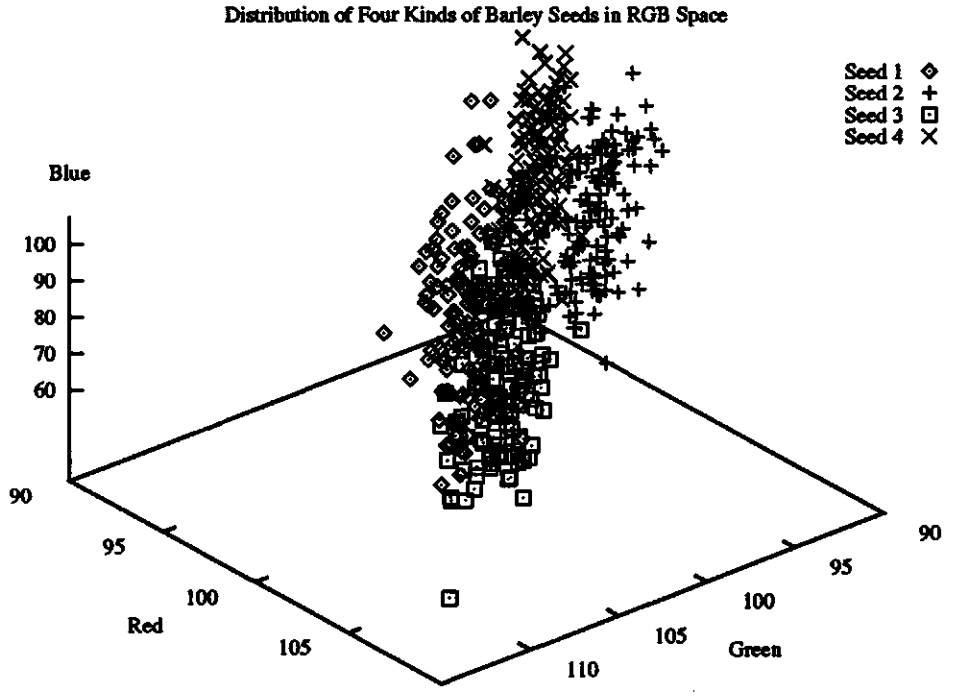


Figure 3.7: 3D plot of neural network input vectors

Table 3.2: The decision rule of the first assignment of output patterns

Tankard	*Breedn*	*Creme*	*CDC Dolly*
$y_{actual} < 0.3$	$0.3 <= y_{actual} < 0.5$	$0.5 <= y_{actual} < 0.7$	$0.7 <= y_{actual}$

number of output neurons was equal to the number of barley classes, i.e. 4 neurons. Each class corresponded to a special output pattern, as shown in Fig 3.8. The desired neuron output value was chosen to be either 0.9 or 0.1 (0.9 as ON and 0.1 as OFF).

When the first assignment of output patterns was applied, classification thresholds were chosen half way between the desired output values. Table 3.2 shows the decision rule for four barley seed varieties, where y_{actual} is the actual output neuron value. In the second assignment of output patterns, the thresholds of each output neuron were set to 0.5. When the output of a neuron was greater than or equal to 0.5, the neuron was at a high activation level and the neuron was said to be turned ON. When the output of a neuron was less than 0.5, the neuron was said to be turned OFF. Therefore, in the actual output pattern, if one neuron was activated to the ON state while all the others were in the OFF state, the seeds should belong to the class of the activated neuron. Other possible situations are considered as undecidable. Table 3.3 shows the decision rule for the second assignment of output patterns.

As mentioned in Section 3.3.1, the maximum number of training epochs of the neural network was 10000. In the training process, after every one thousand epochs the training was stopped to monitor the neural network training status. The testing

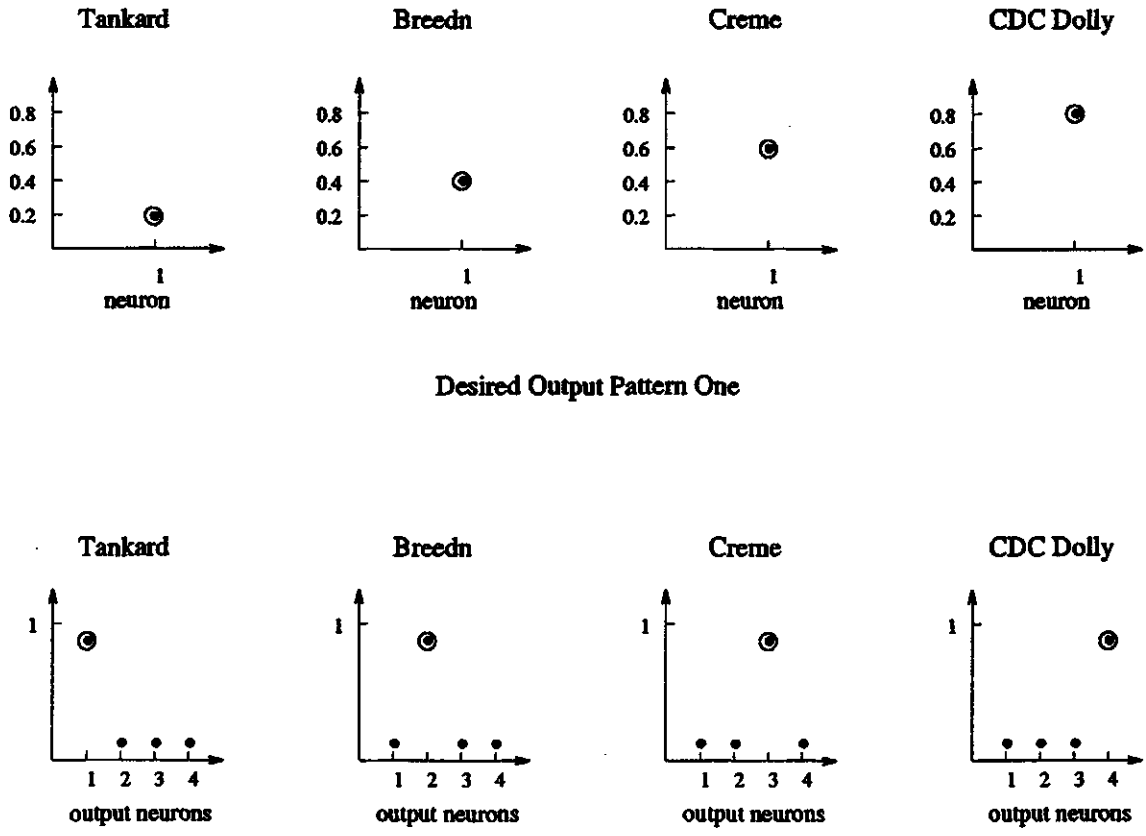


Figure 3.8: Two groups of network output patterns

Table 3.3: The decision rule of the second assignment of output patterns

	Tankard	*Breedn*	*Creme*	*CDC Dolly*
Output Neuron # 1	ON	OFF	OFF	OFF
Output Neuron # 2	OFF	ON	OFF	OFF
Output Neuron # 3	OFF	OFF	ON	OFF
Output Neuron # 4	OFF	OFF	OFF	ON

process was a feedforward only process. The classification accuracy was calculated as a percentage of the test samples that fall within the correct decision boundaries at every one thousand epochs.

3.3.4 Training and Test Data

The number of samples required for successfully training a neural network is affected by many factors, such as the number of input neurons, the number of output neurons, the number of neurons in the hidden layer, the number of hidden layers, and the learning rate. Although there are some experimental results that show how to choose the number of training samples[8], there are no generalized rules applicable to all situations. In this thesis, the size of the training data set was determined first by using a rule of thumb, i.e. using 3 to 5 input/output pairs per feature per class [9]. If the neural network was unable to learn with the given inputs, then the number of input/output vectors was increased until the network could be trained.

The four barley varieties were represented as four classes in the variety discrimination experiments. Two different output patterns were assigned in the experiments. In the training data file, one line contained one training vector consisting of the features of a seed sample and the next line contained the desired output vector corresponding to that seed variety. For each of the four barley varieties, 140 samples were used. This means that there were 1120 lines in the training file where a total of 560 seed

samples were included. The testing data file was created in a similar manner. It also contained 560 seed samples in total, 140 samples from each variety. Different experiments were conducted using different numbers of layers and different numbers of input/output neurons.

3.4 Experimental Results

Several experiments were conducted to obtain acceptable recognition accuracy. During the experiments, different colour features, and different neural network parameters were used for testing. Some typical results are given in this section.

Comparison Between Four Output Neurons and One Output Neuron

In the variety discrimination experiments, the four barley varieties represented four classes. The inputs of the neural network were the colour features of the barley samples. By using the formula described in Chapter 2, the HSI colour features of the barley seeds were transferred from the mean values of their R, G, and B components, which were obtained in the feature extraction phase. The desired output values of the neural network were assigned to each variety. Tankard, Breedn, Creme, and CDC Dolly were assigned to the values of 0.2, 0.4, 0.6, and 0.8, respectively. The neural network structure, referred to as "3-10-1" in Fig 3.9, consisted of 3 inputs, 10 neurons in a single hidden layer, and a single neuron in the output layer. The training curve of

the neural network is shown in Fig 3.9. After 10000 training epochs, the recognition accuracy for Tankard, Breedn, Creme, and CDC Dolly were 75.7%, 68.6%, 67.9%, and 84.3%. The average recognition accuracy of these four barley varieties was 74.1%.

By using 4 output neurons instead of one output neuron, the training curve of the neural network is as shown in Fig 3.10. After 10000 training epochs, the recognition accuracies for Tankard, Breedn, Creme, and CDC Dolly were 72.1%, 73.6%, 75.7%, and 91.4%. The average recognition accuracy of these four barley seed was 78.2%. The average recognition accuracies of 4 output training and 1 output training are shown in Fig 3.11. The average recognition accuracy of the neural network with 4 outputs was higher than the neural network with a single output. The difference of the average recognition accuracies between 4 output neural structure and 1 output neural structure at 1000 epochs was small although it was getting larger until 2000 epochs. For the neural network with a single output, there was no large difference for the recognition accuracy between 1000 epochs and 10000 epochs. The network was trained well at around 1000 epochs. This can be verified from the neural network training curve. As shown in Fig 3.9, the training curve became flat starting around 1000 epochs. If the RMS error was chosen as 0.1, the neural network achieved its training goal at around 1380 epochs. For the neural network with 4 outputs, the recognition accuracy was almost the same between 2000 epochs and 10000 epochs, so the neural network was trained well at around 2000 epochs. This was consistent with

the neural network training curve shown in Fig 3.10. By choosing the RMS error as 0.2, the neural network achieved its training goal at around 3200 epochs.

Different Numbers of Neurons in the Hidden Layer

The number of neurons in the hidden layer affects the neural network training process. If the number of neurons in the hidden layer is too small, a neural network might not be trained well. If the number of neurons in the hidden layer is too large, it may take a long time to train the neural network. A proper number of neurons in the hidden layer is chosen through experiments. Four different neural structures, which had 3, 10, 30, and 50 hidden neurons in one hidden layer respectively, were examined. The average recognition accuracies are shown in Fig 3.12. As can be seen from this figure, the recognition accuracy of the neural structure with only 3 neurons in the hidden layer was much lower than in the other three neural structures with larger numbers of neurons in their hidden layers. The recognition accuracy after 10000 epochs was only 57.8%. The recognition accuracy curves of the other three neural structures were almost the same. The recognition accuracies for neural network structure "3-10-4", "3-30-4", and "3-50-4" after 10000 epochs were 78.2%, 78.8% and 78.4%, respectively. The training time of the neural structure "3-50-4" was about 6 times that of the neural network structure "3-10-4". Increasing the number of neurons in the hidden layer does not affect the neural network training much but

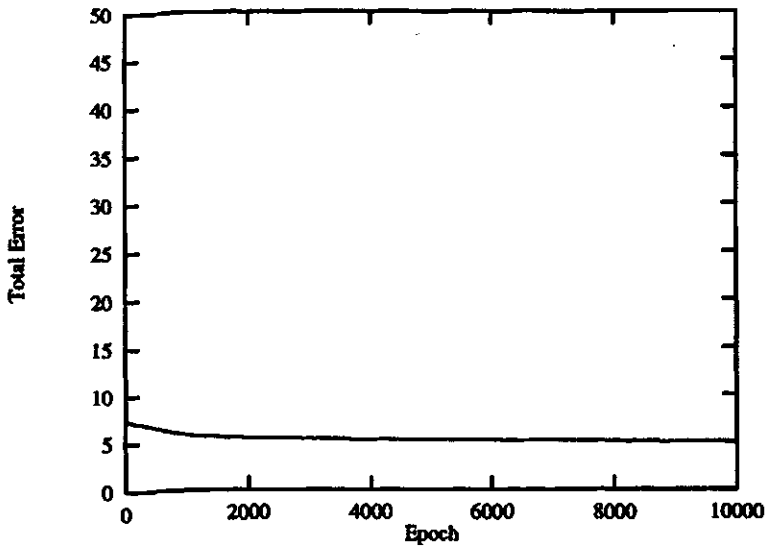


Figure 3.9: Training curve of 4 varieties using a neural structure 3-10-1

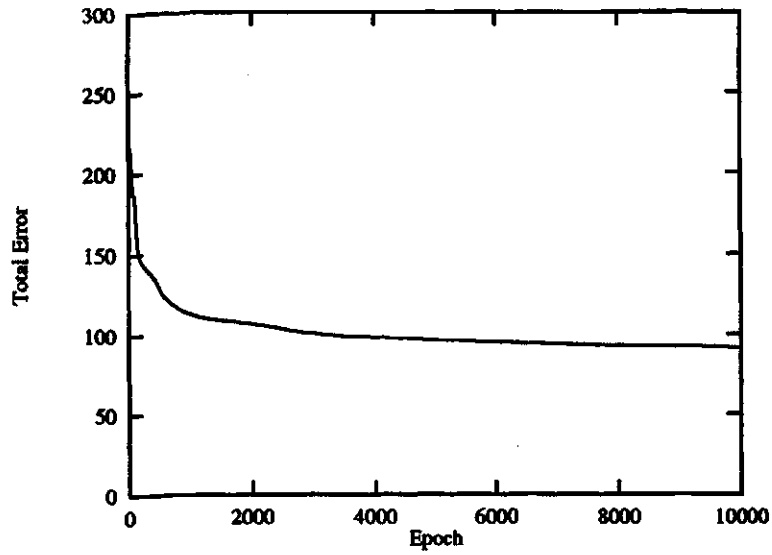


Figure 3.10: Training curve of 4 varieties using a neural structure 3-10-4

it requires a much longer time to train the larger neural network.

Different Number of Hidden Layers

Training of a neural structure with two hidden layers was studied to try to increase the recognition accuracy. A number of trials were conducted and different numbers of neurons were chosen. The learning parameter μ was varied in the range between 0.01 and 0.2. To balance the accuracy and the training time, the best result was obtained by choosing μ as 0.01 and a neural structure of "3-20-20-4", which consisted of 3 inputs, 10 neurons in the first hidden layer, 10 neurons in the second hidden layer, and 4 neurons in the output layer. The recognition accuracy of four barly varieties were 64.3%, 83.6%, 92.1%, and90% respectively. The average recognition accuracy of 82.7% was achieved after 10000 training epochs. The recognition accuracy curves of this neural structure and a neural structure with a single hidden layer are shown in Fig 3.13. The sum of squared error curves are shown in Fig 3.14. After 3000 training epochs, values of the recognition accuracy of both neural networks were 80.9% and 76.6% respectively. The neural structure with two hidden layers was somewhat better than the neural structure with a single hidden layer in the barley seed classification.

Different Colour Features

In this experiment, different colour features were used as the neural network inputs. Four barley varieties, Tankard, Creme, Breedn, and CDC Dolly, were used as

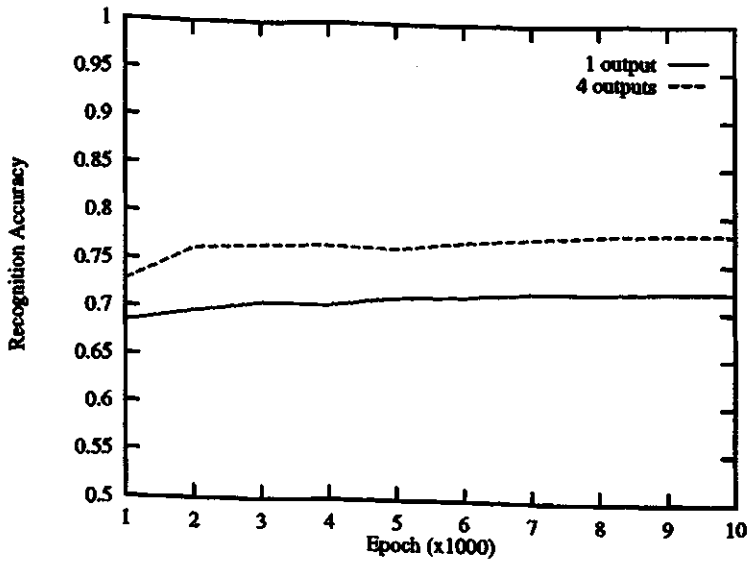


Figure 3.11: Recognition accuracy of 4 varieties using a neural structure 3-10-1 and 3-10-4

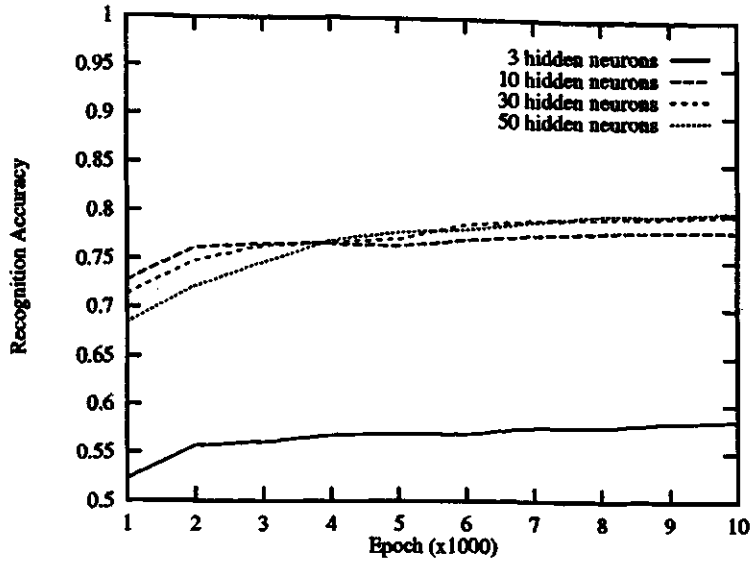


Figure 3.12: Recognition accuracies of 4 varieties using neural structure with different number of neurons in the hidden layer

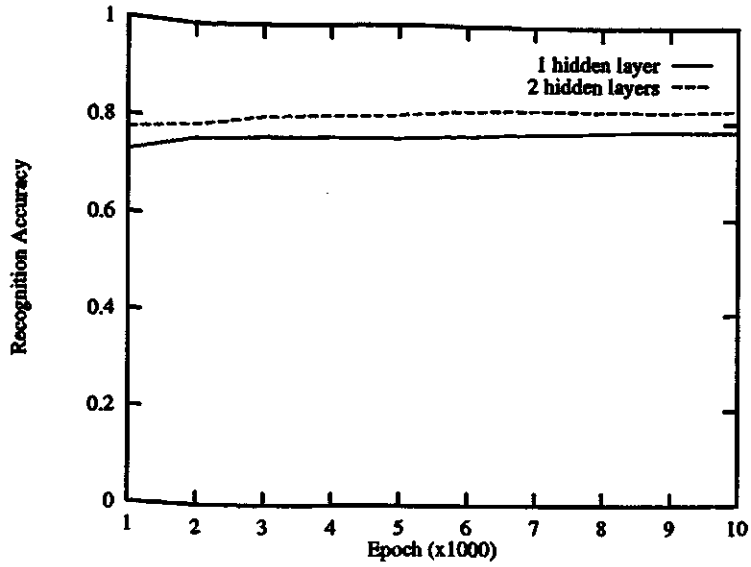


Figure 3.13: Recognition accuracy of 4 varieties using neural structures with different numbers of hidden layers

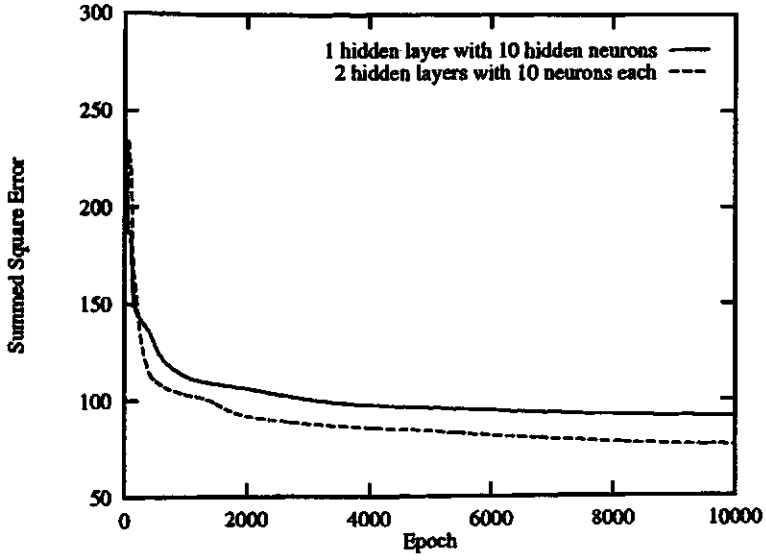


Figure 3.14: Training curves for classification using neural structures with different numbers of hidden layers

test samples. For each variety, 280 samples were measured, i.e. 140 for training data and 140 for testing data. The learning rate was from 0.01 to 0.2. Reasonable results were obtained in a 3-10-10-4 neural network structure by using a learning rate of 0.01.

The colour features were transformed from RGB space to HSI and Lab spaces. The classification accuracies based on different colour spaces are shown in Table 3.4.

Table 3.4: The classification accuracies of a four variety classification using a neural network structure of 3-10-10-4 based on different colour spaces

	Tankard	*Breedn*	*Creme*	*CDC Dolly*	*Average*
RGB Colour Space	55.7%	65%	75%	90.7%	71.6%
Lab Colour Space	75.7%	83.6%	77.1%	78.6%	78.8%
HSI Colour Space	65%	77.1%	90%	90%	80.5%

The results show that the highest recognition accuracy was obtained with colour features represented in HSI colour space. The Creme seeds and CDC Dolly seeds got the highest recognition accuracy, 90%. Tankard was the lowest at 65%. Breedn was in the middle at 77.1%. The curves of recognition accuracy vs. training epochs using HSI colour features are shown in Fig 3.15. The neural network training curve using HSI colour features is shown in Fig 3.16. According to this training curve, the training goal could be chosen as 76. The average RMS error was 0.18.

3.5 Summary

Barley seed classification using image analysis techniques and neural network were studied. A machine vision system was used to capture the colour image of barley

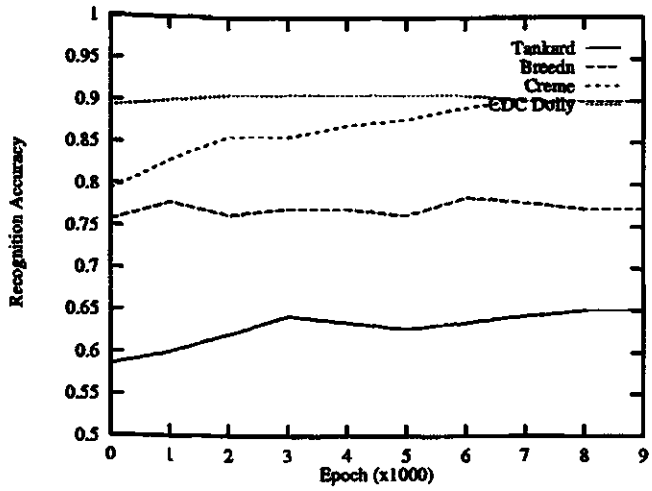


Figure 3.15: Recognition accuracy curves of a 4 variety classification using a 3-10-10-4 neural structure

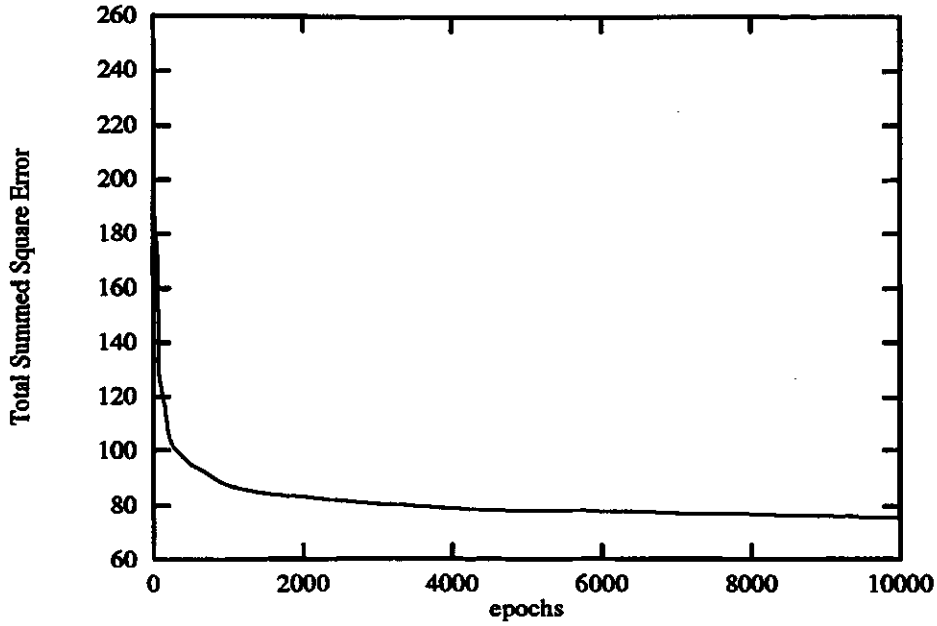


Figure 3.16: The training curve of a 3-10-10-4 neural structure using HSI colour features

seeds. Based on the image histogram, an object isolation method was implemented to isolate barley seeds from the background. Different colour features were used to classify different kinds of barley seeds. According to the KL transform of the colour features and the experimental results, HSI was the best choice for representing the colour features of the barley seeds. A multi-layer neural network was used as a classifier. Different neural structures were tested to achieve better recognition accuracies.

Experimental results for barley seed classification based on their colour features were presented in this chapter. Four barley varieties, Tankard, Breedn, Creme, and CDC Dolly, were tested. 140 samples for each seed variety were used as a training test set. The same number of samples were used for testing.

Different neural structures were tested to compare their results. Comparing with a one output neural structure, a four output neural structure achieved better results. The recognition accuracies after 10000 training epochs for Tankard, Breedn, Creme, and CDC Dolly were 72.1%, 73.6%, 75.7%, and 91.4%, respectively. The number of hidden neurons affects the neural network training as well. If there are too many neurons in the hidden layer, neural network training takes a long time. If there are too few neurons in the hidden layer, the neural network may not be adequately trained. A reasonable result was obtained by using a neural structure of "3-10-4". The average recognition accuracy was 78.2% and the summed squared error was 91.7. The RMS error was 0.211. The neural structure with two hidden layers was tested against the

single hidden layer neural structure. The test result showed that the two hidden layer neural structure produced better results. The average recognition accuracy was 82.7%. The colour features represented in different colour spaces, such as RGB, Lab, and HSI, as the neural network input were also tested. The best result was obtained by using HSI colour features. Among all the experimental results, the best result for the four barley seed classification was achieved by using HSI colour features as the inputs, and the neural structure of "3-20-20-4", i.e. the neural network which had 3 inputs, 4 outputs, and 2 hidden layers with 20 neurons in each hidden layer. The average RMS error in this case was 0.182. The average recognition accuracy was 82.7%.

Chapter 4

Lossless Image Data Compression

4.1 Introduction

The objective of lossless image coding is to represent an image with as few bits as possible with the condition that any degree of image degradation is not allowed. Image data contain significant structure since pixel-to-pixel correlation exists in images. Efficient coding can be accomplished by first determining the structure of the data and then developing encoding algorithms which are efficient for that data structure. Generally, there are two kinds of superfluous information contained in an image. The first one is *subjective redundancy* and the second one is *statistical redundancy*. Subjective redundancy is based on the human visual system. It can be removed intentionally without causing a loss in its subjective quality. The removal of subjective

redundancy is irreversible so that the original image data cannot be recovered after the removal of subjective redundancy. Statistical redundancy is based on image data statistics. It can be removed from the data without destroying any information, and the original data can be recovered. Statistical redundancy can be considered to occur in two types. The first type occurs when the autocorrelation function of an image is not a delta function (uncorrelated). The second type occurs when the probability density function of the signal is not uniform. In image coding, redundant information need not be coded if the redunancy is removed. In order to make coding more efficient, these two types of statistical redundancies should be removed as much as possible. Lossless coding generally means that the image received is the image sent in terms of the pixel value of the image, not in terms of the visual impression of the image. With this constraint, only statistical redundancy can be removed.

In this chapter, two methods will be studied in an attempt to reduce statistical redundancy. Algorithms for constructing differential images will be introduced. Combining redundancy removal algorithms and a modified Huffman coding, the two types of statistical redundancy are reduced. The implementation of modified Huffman coding is described and the evaluation of it is presented. The compression gain of different types of encoding algorithms are evaluated.

The encoding and decoding sequence is shown in Fig 4.1. Two opposite processes of encoding and decoding are included in Fig 4.1. The sequence is divided in four

steps: constructing the differential image, encoding the image by using modified Huffman coding, decoding, and reconstructing the image. The algorithms for constructing a differential image were evaluated by applying the algorithms to several images. The efficiency of modified Huffman coding was also evaluated in terms of actual bit rate.

4.2 Algorithms for Constructing Differential Images

It is well known that pixel-to-pixel correlation exists in images [20]. Pixel difference coding is an efficient method to reduce this correlation. For the purpose of removing the correlation between adjacent pixels and also making the recovery of the original image possible, two algorithms of constructing difference images are introduced.

4.2.1 Difference Image

Let us denote an image by A. z^{-1} represents a one-step shift operator towards the positive side of the horizontal axis (or vertical axis), then the first difference image D_1 could be expressed as:

$$D_1 = A - z^{-1}A = (1 - z^{-1})A \tag{4.1}$$

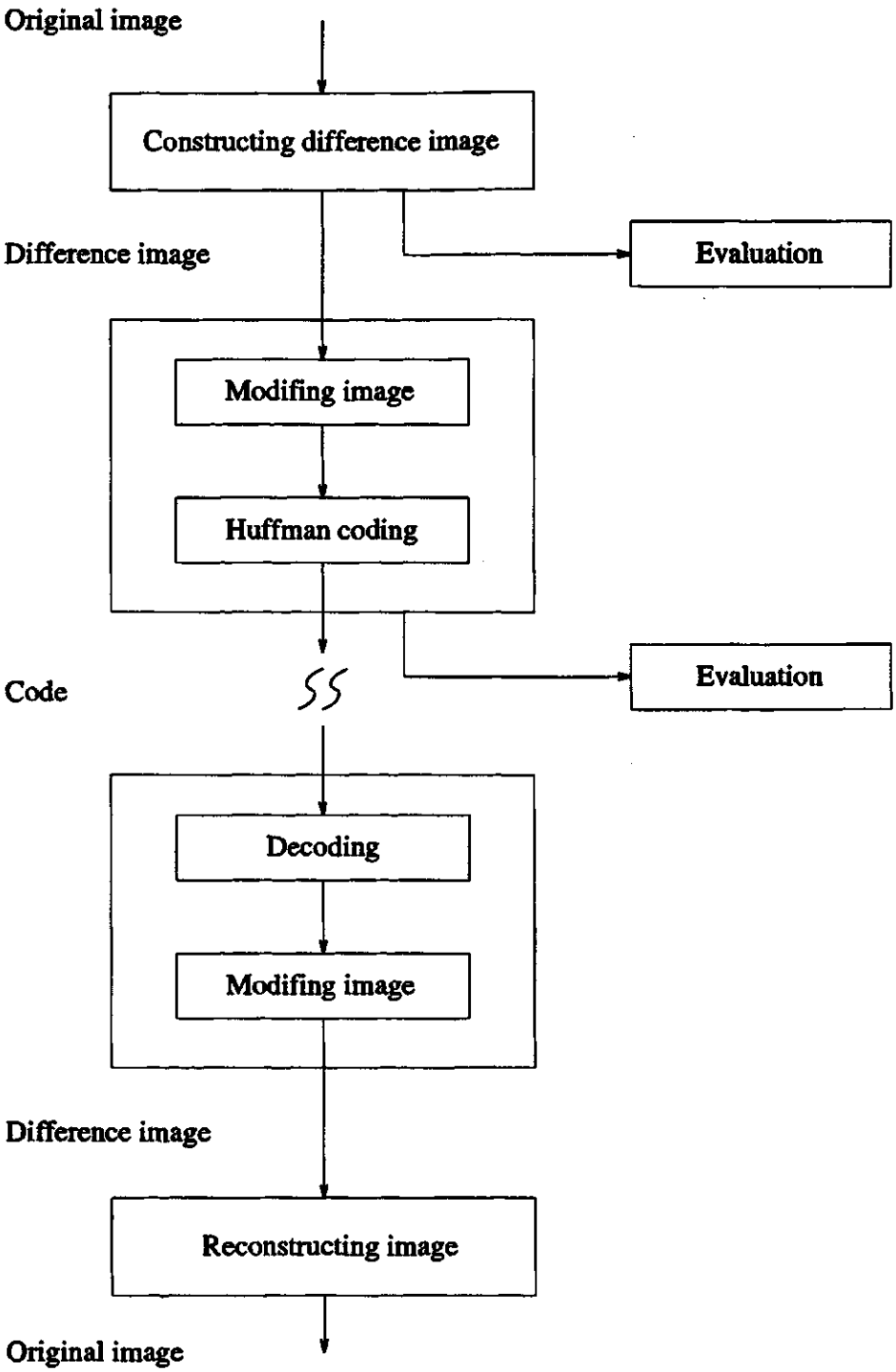


Figure 4.1: The organization of the experiments

the second difference image D_2 is given by:

$$D_2 = D_1 - z^{-1}D_1 = (1 - z^{-1})^2 A \tag{4.2}$$

and the n-th difference image D_n is given by:

$$D_n = D_{n-1} - z^{-1}D_{n-1} = (1 - z^{-1})^n A \tag{4.3}$$

A 4×4 image and the first difference image obtained by shifting the original image in the horizontal direction are shown below, where $x_{i,j}$ denotes a pixel in the image, i is the pixel index in the horizontal axis and j is the pixel index in the vertical axis.

$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	$x_{1,4}$
$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	$x_{2,4}$
$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	$x_{3,4}$
$x_{4,1}$	$x_{4,2}$	$x_{4,3}$	$x_{4,4}$

A 4×4 image

$x_{1,1}$	$x_{1,2} - x_{1,1}$	$x_{1,3} - x_{1,2}$	$x_{1,4} - x_{1,3}$
$x_{2,1}$	$x_{2,2} - x_{2,1}$	$x_{2,3} - x_{2,2}$	$x_{2,4} - x_{2,3}$
$x_{3,1}$	$x_{3,2} - x_{3,1}$	$x_{3,3} - x_{3,2}$	$x_{3,4} - x_{3,3}$
$x_{4,1}$	$x_{4,2} - x_{4,1}$	$x_{4,3} - x_{4,2}$	$x_{4,4} - x_{4,3}$

First difference image

To make image reconstruction possible, the first column of the first difference image retains the first column of the original image. For the same reason, the second difference image keeps the first column of the first difference image as its first column. The sizes of the difference images are the same as the original one.

Successful applications of the difference operation remove the pixel-to-pixel correlation. The more applications applied, the more random the resulting image become.

4.2.2 Hierarchy Embedded Differential Images

Hierarchy embedded differential image (HEDI) is another type of pixel difference image coding method [6]. The advantage of HEDI is that image quality progressively improves as image transmission progresses. Difference images are constructed by using sub-sampling or mean-sampling methods for the predictor with block sizes of 2×2 or 3×3. One method, HEDI-CS2 (which is HEDI in circular direction with sub-sampling of block size 2x2), was implemented and applied to MRI images. The method of constructing HEDI-CS2 is shown in Fig 4.2.

The construction of HEDI-CS2 is described as following. X_L denotes an $N \times N$ image. $\{X_l\}$ denotes a set of image frames in a pyramid form, where l denotes the level number, level 0 corresponds to the lowest resolution frame and level L corresponds to the highest resolution frame, i.e. original image. For an 8x8 image, l is from 0 to 3. The number of pixels in the image frame X_l is $2^l \times 2^l$. As shown in Fig 4.2,

the lowest resolution frame, X_0, is a shaded pixel, X_1 has four shaded pixels, X_2 has sixteen shaded pixels, and the highest resolution frame, X_3, is the original 8x8 image. The difference image, denoted as Y_L, should have the same size as the original image.

For building HEDI-CS2, level 0 is considered first. At level 0, let image frame X_0, i.e. the shaded pixel in Fig 4.2(a), be $x_{0,0}$. The first element of the difference image Y_L, $y_{0,0} = x_{0,0}$. At level 1, image frame X_1 consists of four shaded pixels as shown in Fig 4.2(b), where $x_{0,0}$ is the parent node and the other three are children nodes. The errors, which are the difference between two nodes, are computed circularly and stored in Y_L. i.e.

$$y_{4,0} = x_{0,0} - x_{4,0} \tag{4.4}$$

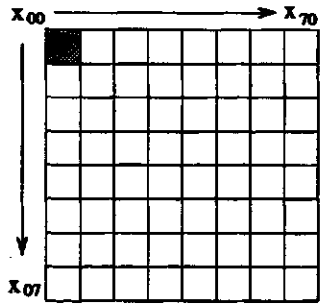
$$y_{4,4} = x_{4,0} - x_{4,4} \tag{4.5}$$

$$y_{0,4} = x_{4,4} - x_{0,4} \tag{4.6}$$

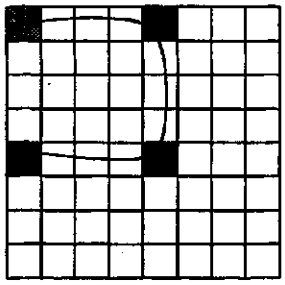
Note that only three children nodes in Y_L are computed.

Generally, at level l, X_L is divided into 2^{2l} sub-images of size $2^{L-l} \times 2^{L-l}$, and the errors at the child nodes of each quadrant are computed with their root node and stored in Y_L. The direction of taking differences for errors are shown in Fig 4.2(e). The errors are computed by:

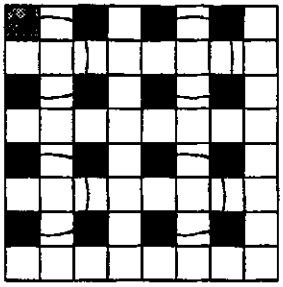
```
for(i=0; i<N; i=i+2*s)
{
    for (j=0; j<N; j=j+2*s)
    {
        y[i+s][j] = x[i][j] - x[i+s][j];
        y[i+s][j+s] = x[i+s][j] - x[i+s][j+s];
```



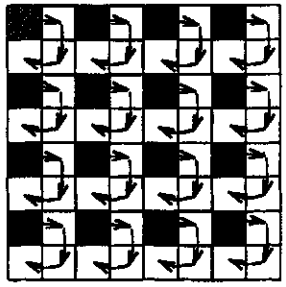
(a) Level x_0



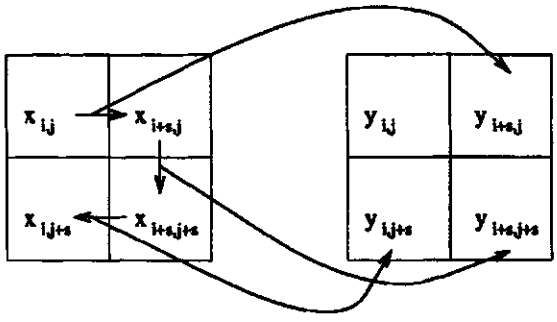
(b) Level x_1



(c) Level x_2



(d) Level x_3



(e) Direction of error calculation

Figure 4.2: 2-by-2 method on an 8x8 image and error calculation direction

```
        y[i][j+s] = x[i+s][j+s] - x[i][j+s];
    }
}
```

while $s = 2^{L-l}$, and N is the number of pixels in each row or column. Eventually all pixels except the root node in y_L are the errors with their parent nodes.

The difference image constructed in this way can be used to recover the original image. Reconstructing the original image is a reverse process of the above algorithm. The general algorithm for level l is as follows:

```
for(i=0; i<N; i=i+2*s)
{
    for (j=0; j<N; j=j+2*s)
    {
        x[i+s][j] = y[i][j] - y[i+s][j];
        x[i+s][j+s] = y[i+s][j] - y[i+s][j+s];
        x[i][j+s] = y[i+s][j+s] - y[i][j+s];
    }
}
```

The reconstruction of an image can be performed level by level. The original image is completely recovered when level L is reached.

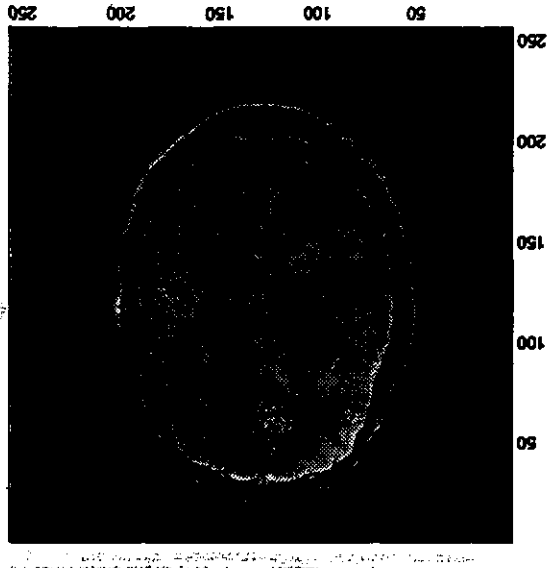
4.3 Algorithm evaluation and Analysis

In this section, the algorithms discussed in the previous section were evaluated by applying these algorithms to some gray scale MRI human head images of size 256 $\times$ 256 with 8 bits per pixel. The evaluations were made based on the calculated

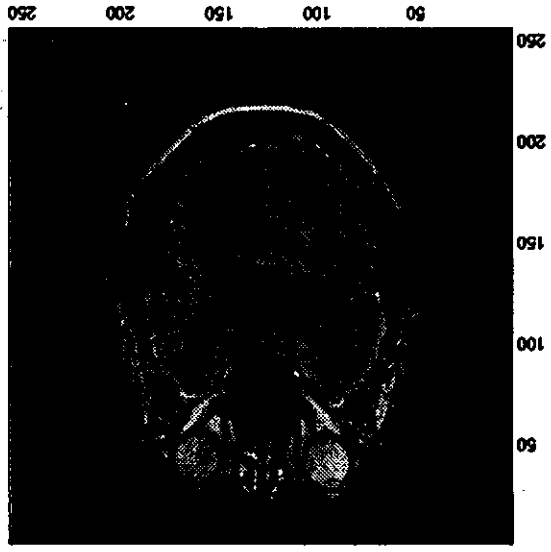
entropy of the difference images, and the results were also analyzed according to auto-correlation functions of the difference image.

As mentioned in chapter two, the average information over an image is represented by the entropy. The entropy of an image measured in bits/pixel indicates how many bits could be removed per pixel as redundancy. The first order entropy is based on the first-order probability density function of the image. The pixel values of the image are treated as random variables. If eight bits are used to code an image and the probability of any number between 0 and 255 occurring is the same, the information entropy is 8. If only one value, say 200, occurs all times, the entropy is zero. The entropy of an image is a function of its probability density function, which is estimated from the image histogram. For a uniform histogram, the entropy is maximum. For a histogram having a concentrated peak, the entropy is small. The narrower the peak is, the smaller is the entropy. In calculating the first order entropy, the correlation of adjacent pixels is not taken into account.

Algorithms for constructing difference images were evaluated by applying them to some slices of a 3-D MRI human head section image and the commonly used testing image "Lena". The 3-D MRI head section image consists of 64 slices. Fig 4.3 shows two slices of this image set. Each of these two images consists of 256 x 256 pixels and they are all 8 bit gray scale images. Fig 4.4 shows the "Lena" picture, which is a 512 x 512 pixel and 8 bit gray scale image.



(a)



(b)

Figure 4.3: Two slices of a 3-D MRI human head image

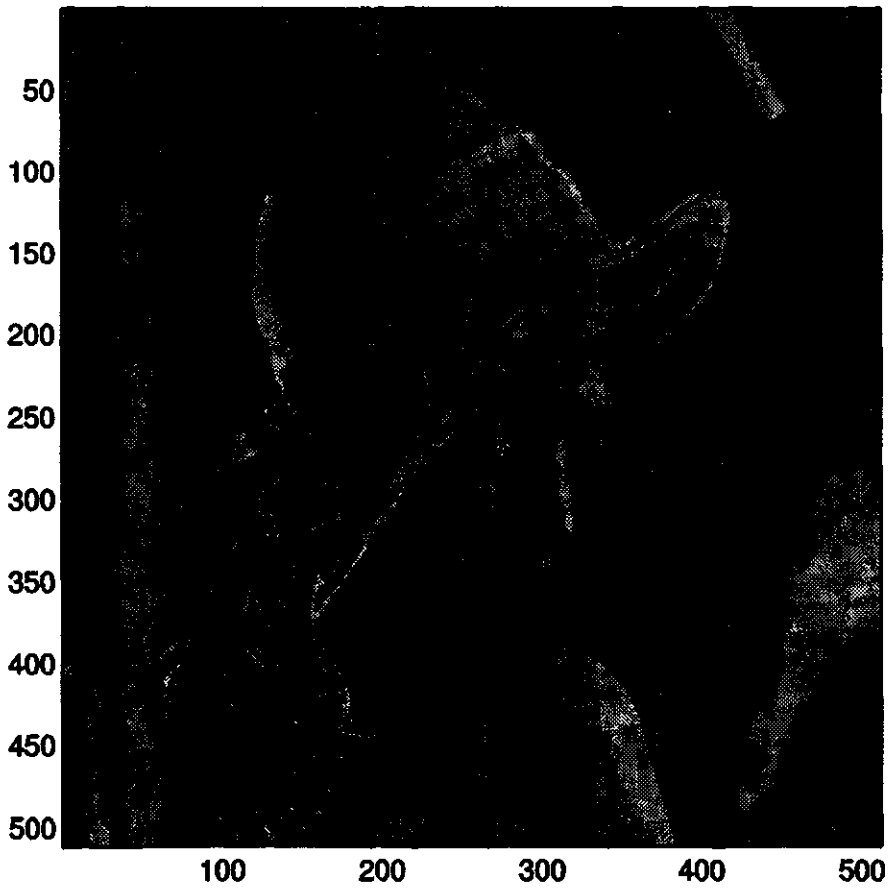


Figure 4.4: 512 x 512 8 bits gray scale Lena image

4.3.1 The evaluation of standard difference image algorithm

The algorithm for constructing a standard pixel difference image was applied to the two MRI head images shown in Fig 4.3 and the image "Lena" shown in Fig 4.4. Difference images were obtained by shifting the original image step by step in the horizontal direction. First order entropies were calculated up to their n-th difference images based on their probability density functions (PDF's). Shifting was applied only in the horizontal direction. Fig 4.5 shows the probability density functions of the head image shown in Fig 4.3(a), and the probability density functions of the first and second difference images. It is observed that the probability density function of the original image has two peaks. The probability density function of the difference images has only a single peak. The peak value of the PDF of the first difference is larger than that of the second difference and the width of the peak is narrower than the second one. This can be explained by observing the difference image data structure. The first difference image was constructed by shifting the original image in the horizontal direction and then taking the difference between the shifted one and the original one. Since the original image was highly correlated, a high peak around 0 in the PDF of the first difference image was found. The first application of pixel difference calculation results in a well uncorrelated image. Further difference calculation to obtain a higher difference image D_n disperses the difference values; the PDF was thus flattened.

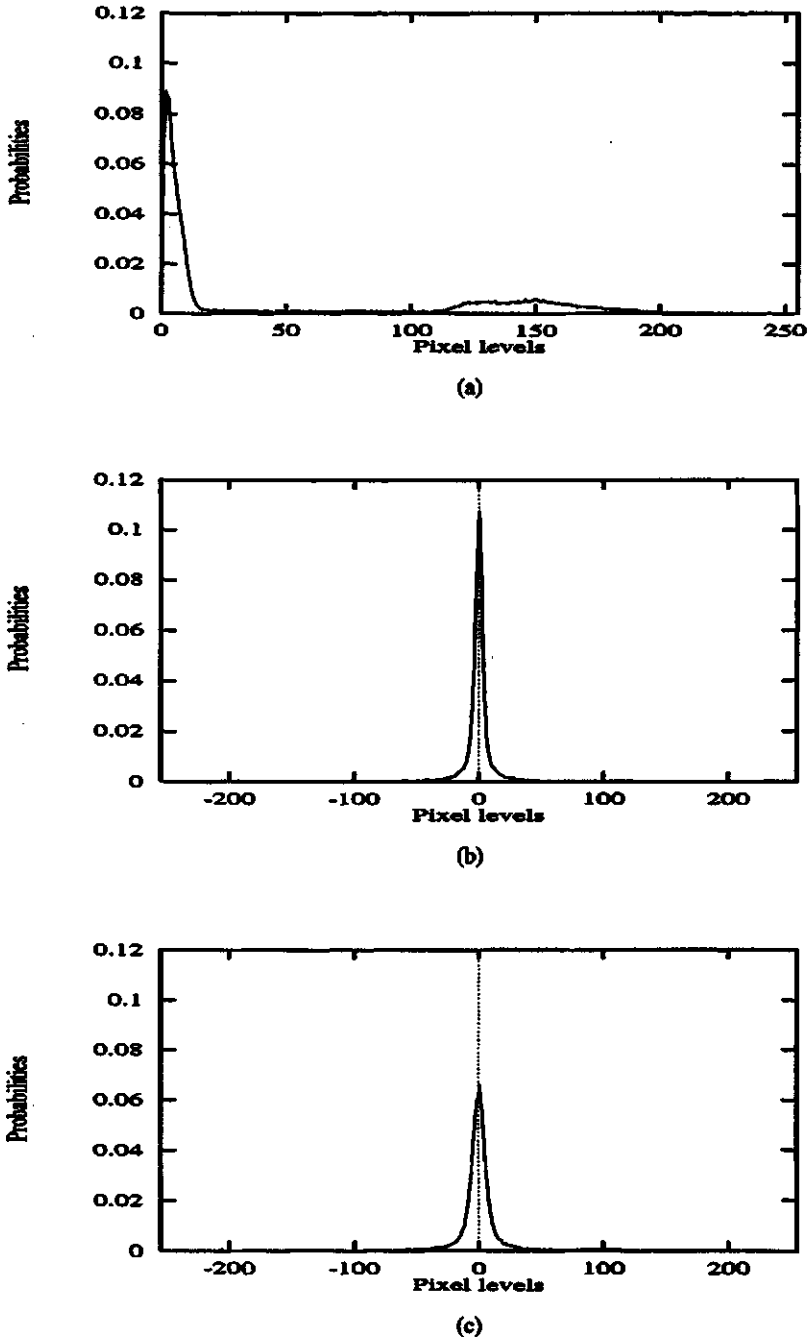


Figure 4.5: The PDFs of difference images (shifting in the horizontal direction) of head image, Fig4.3(a) where (a) The PDF of original image (b) The PDF of the first difference image (c) The PDF of the second difference image

For MRI head images and the image Lena, Fig 4.6 shows the variety of entropy values with the difference order. From this figure we can see that the first difference image has the lowest entropy compared with the original image or higher difference images. Table 4.1 gives the calculated entropy values, where $n = 0$ denotes the original image, $n = 1$ denotes the first difference image, etc. For those two MRI head images, the entropies of first difference images are reduced by 0.75 and 0.77 bits/pixel respectively from those of the original images. For the Lena image, the entropy is reduced by 2.74 bits/pixel. According to these results, for the best compression, the operation for constructing a difference image should be applied to the original image only once.

Table 4.1: Entropies (bits/pixel) of higher order difference images D_n(shifting in horizontal direction)

Difference Image	*Entropy*		
	image head(a)	*image head(b)*	*image Lena*
n=0	5.615405	5.928356	7.445505
n=1	4.861321	5.157103	4.704742
n=2	5.448454	5.702962	5.122688
n=3	6.240144	6.460389	5.842466
n=4	7.103492	7.307552	6.674747
n=5	7.997103	8.191278	7.553031
n=6	8.908238	9.094923	8.458877
n=7	9.813683	10.000862	9.380965
n=8	10.725728	10.903172	10.312697
n=9	11.610693	11.789503	11.248829
n=10	12.468448	12.637448	12.182171

The same entropy calculations were made for the MRI head and "Lena" difference

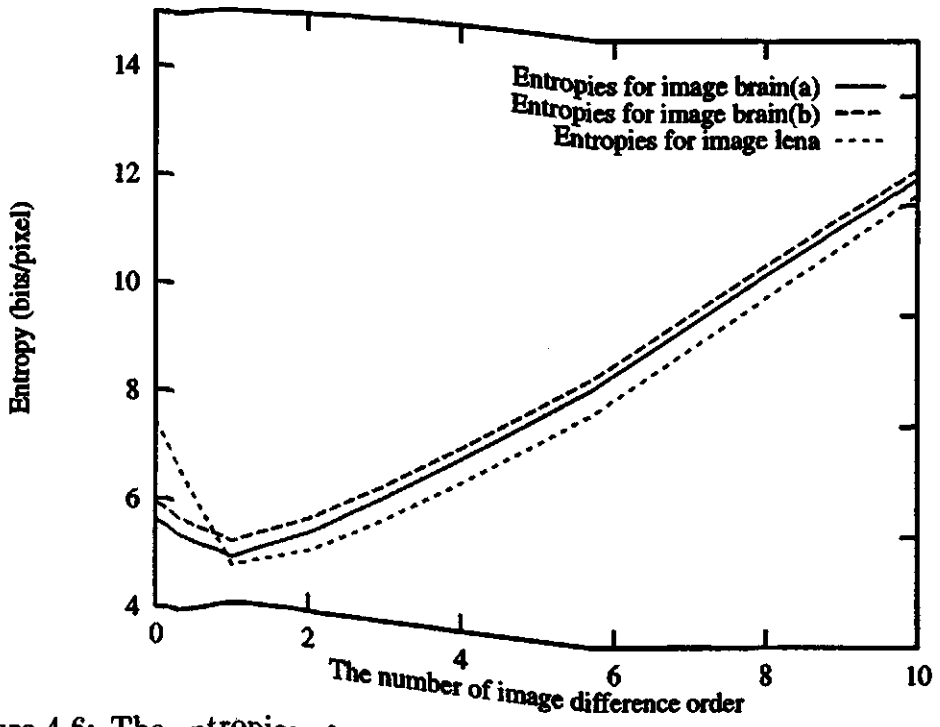


Figure 4.6: The entropies of difference image(shifting in horizontal direction)

images obtained by shifting original images in the vertical direction. The results are shown in Table 4.2. The entropy of the difference image in vertical shifting is larger than that of the entropy of the difference image in horizontal shifting. For all n-th difference images, the least entropy is still the entropy of the first difference image.

Table 4.2: First entropies (bits/pixel) of difference images(shifting in vertical direction)

Difference Image	*Entropy*		
	image head(a)	*image head(b)*	*image Lena*
n=0	5.615405	5.928356	7.445505
n=1	5.011084	5.374353	5.089800
n=2	5.666790	5.987724	5.433792
n=3	6.479097	6.767675	6.148915
n=4	7.353111	7.629685	6.982970
n=5	8.253713	8.520482	7.860569
n=6	9.161461	9.161461	8.766186
n=7	10.064533	10.064533	9.686005
n=8	10.964142	10.964142	10.615208
n=9	11.833531	11.833531	11.545094
n=10	12.664618	12.664618	12.470309

4.3.2 The evaluation of HEDI-CS2 algorithm

Images shown in Fig 4.3 and Fig 4.4 were also used to test the HEDI-CS2 algorithm. The probability density function of MRI head image (a) was plotted in Fig 4.7. Comparing this figure with the PDFs of the standard pixel difference images in Fig 4.5, this graph is similar to that obtained from the standard pixel difference image.

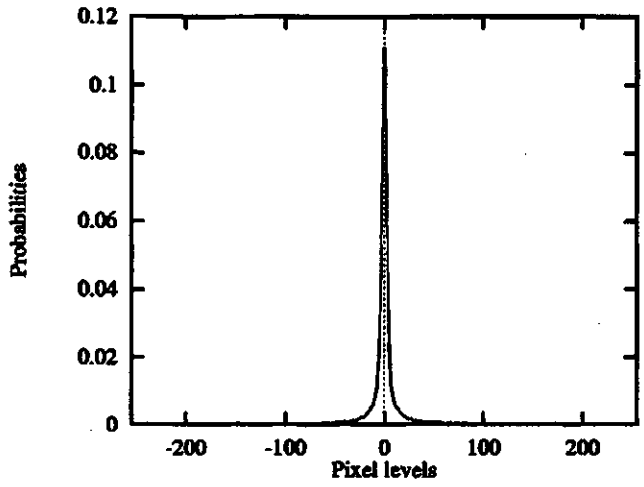


Figure 4.7: The PDF of HEDI-CS2 of MRI brain image(a)

Table 4.3: First entropies (bits/pixel) of HEDI-CS2 images

	image head(a)	*image head(b)*	*image Lena*
Entropies of original images	5.615405	5.928356	7.445505
Entropies of HEDI-CS2 images	5.078765	5.421124	5.002786

Table 4.3 shows entropies of the constructed images. For MRI head images, entropies decreased to 5.08 bits/pixel and 5.42 bits/pixel, respectively, from original images in HEDI-CS2 images. For image Lena, the entropy decreased to 5.00 bits/pixel. Comparing these results with the results obtained by using standard difference method, there is no significant difference for calculated entropies. The HEDI-CS2 method is better in the sense that it allows progressive transmission of images.

Progressive transmission of an image allows the initial reconstruction of an approximation followed by a gradual improvement of quality in subsequent image reconstruction. It is particularly important for browsing large image files in a picture archive or reconstructing an image on the fly while the image data is being transmitted. The usefulness of a progressive transmission scheme depends upon how soon one can recognize the image from the levels transmitted. The lesser the amount of information required to recognize an image, the better the scheme is, because the lossless image coding can be turned into a lossy image coding for some images if the hierarchical process is stopped at a level lower than the top level. The sequence of image reconstruction for a MRI head image (a) is shown in Fig 4.8, using 2 x 2 blocks. The reconstruction levels from 3 to L are shown, where L equals to 8.

4.3.3 Summary

Based on the first order entropy of the difference images, two algorithms of constructing difference images were evaluated. Test results showed that the standard pixel difference image shifted in the horizontal direction had the smallest entropy. The HEDI-CS2 images had larger entropies, but the data structure of these images illustrates how the image in progressively improved while the reconstruction is carried out.

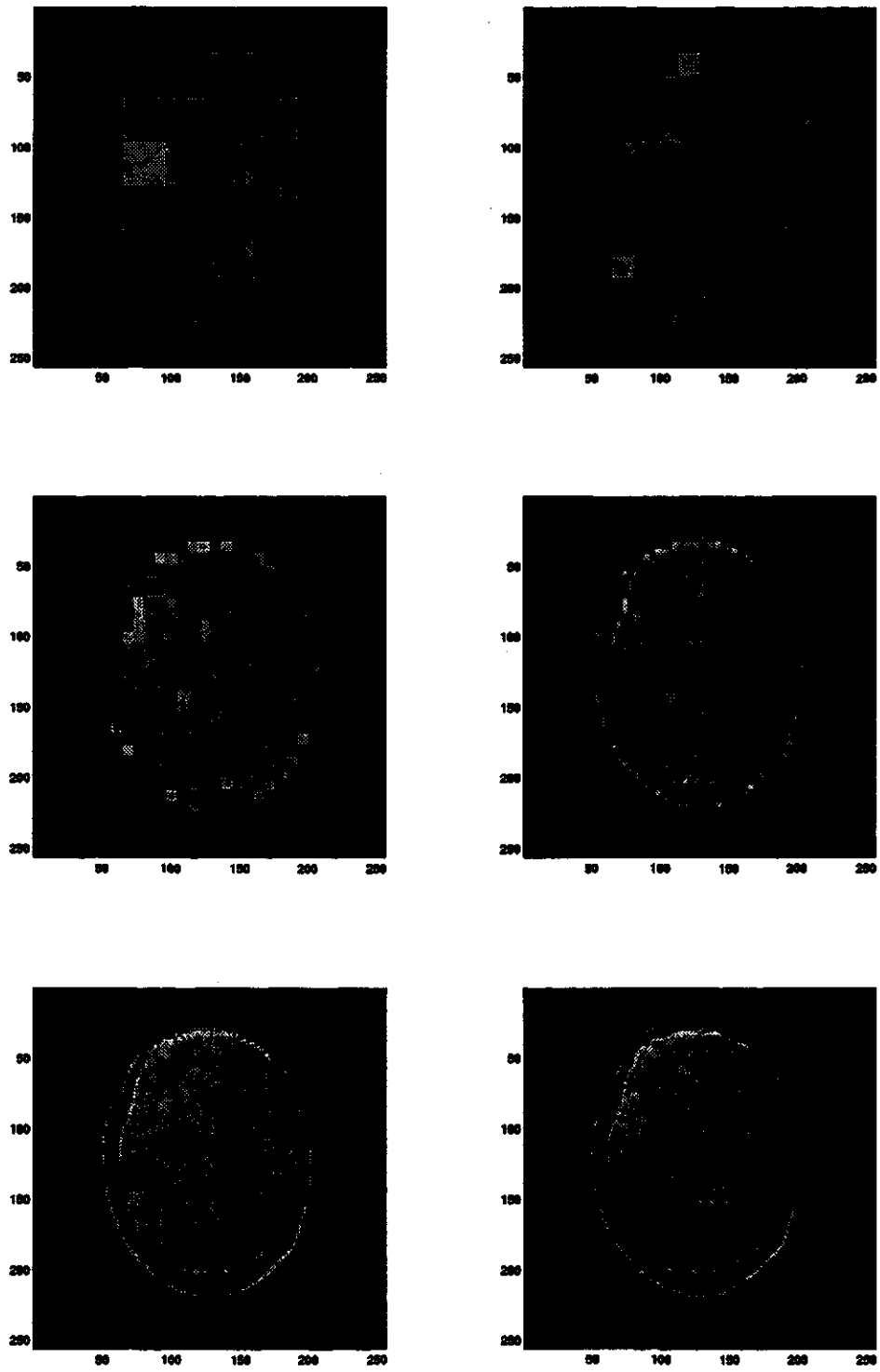


Figure 4.8: Reconstruction of the MRI head image(a) with 2 x 2 block

4.4 Second order entropy and modified Huffman Coding

As shown in the previous section, image redundancy can be reduced by constructing difference images. The entropy of the first difference image is less than that of the original image. Since difference image constructing algorithms allow complete image reconstruction of the original image, the difference image can be coded and transmitted as a replica of the original image.

Second order entropy

The first order entropy was calculated to estimate the information content of the image in the last section. For a more accurate estimation, the second order entropy can be employed. The dependency between adjacent pixels is taken into account in calculating the second order entropy. Considering the adjacent pixels in the horizontal direction from left to right, one pixel has a value i and the adjacent pixel has a value j; the joint probability of the pixel pair which has the value of (i, j), is represented in Equation 2.33. For example, in the MRI head image (a), image size is 256×256. The total number of pixel pairs equals to image size minus 1, which is 65535. The number of pixel pairs (i, j) is the number of pixels in which its own pixel value is i and the adjacent second pixel value is j. The second order entropy is calculated based on the joint probability. The calculation equation is given in Equation 2.35.

Table 4.4: First and second order entropies (bits/pixel) of MRI image(a)

Entropy	*Original image*	*First difference image*
First order entropy	5.615445	4.861343
Second order entropy	4.741966	4.612613

Using the MRI head image (a) as an example, second order entropies of the original image and the standard pixel difference image were calculated. The results are shown in Table 4.4.

Comparison reveals that the second order entropies are smaller than the first order entropies. Table 4.4 also shows that the entropies calculated for the first difference images are smaller than those of the original images. This implies that there is still a pixel-to-pixel correlation in the first difference image. This kind of redundancy is not removed by using algorithms of constructing a difference image. Another method, which takes the second order entropy into account, and Huffman coding, was tried to further reduce this redundancy.

Modified Huffman coding

Huffman Coding is an optimal coding method in the sense of that it minimizes the average bit rate. Huffman Coding is based on the information entropy of the image data. The calculation of the joint entropy is based on the joint probability density of adjacent pixels. To remove more redundancy, the dependence between adjacent pixels

must be considered when an image is coded. A method that modifies the Huffman coding to include adjacent pixel correlations is proposed and implemented.

Let us start with the joint probability density of an image. For the joint probability density of an image, two adjacent pixels are considered together. The joint probability density function of the standard pixel difference image of MRI head image (a) is shown in Fig 4.9. As shown in this figure, the peak of the probability density is around (0,0), i.e. both the first and the second pixels take value of 0. This shows that in the first difference image, the pixel pairs which took values around (0,0) appeared most frequently. The frequently appearing pixel pairs, such as (0,0), are called a mode. Before an image is coded, frequently appearing modes are searched. In Huffman coding, those modes are assigned to some special symbols to integrate the second order PDF into the first order PDF. In the experiments, the code assignment consists of two procedures: image size shrinking and Huffman coding, where image size shrinking utilizes the result of the mode search.

Image Size Shrinking

The size of an image equals the number of pixels in a row multiplied by the number of pixels in a column. The size of the MRI head image, for instance, is 256 X 256 pixels. The total volume of an image, i.e. the total bit number of the image, is defined as

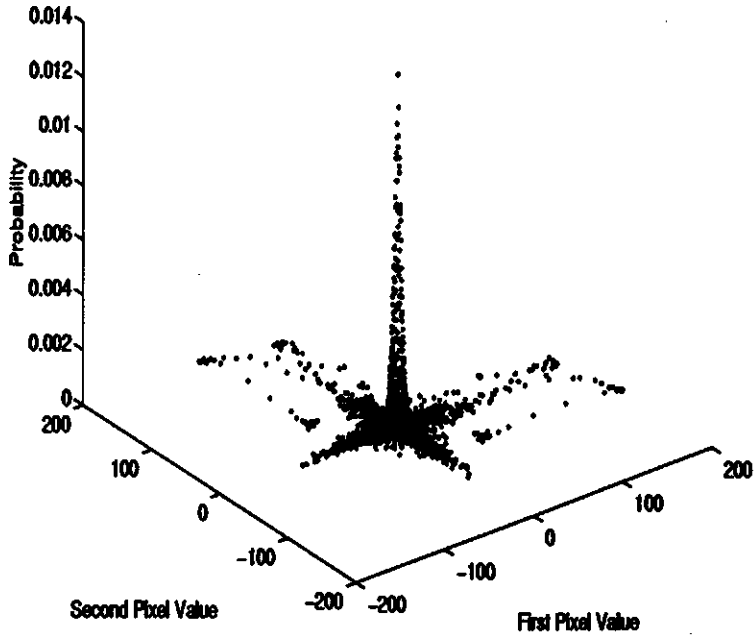


Figure 4.9: The joint PDF of the first difference image of MRI head image(a)

$$Total\ Volume = image\ size \times bits\ per\ pixel \tag{4.7}$$

In the image size shrinking step, each frequently appearing mode is represented by a special value (symbol), which does not occur in the image. Since each mode represents two adjacent pixels in one symbol, the size of the image is reduced.

The first difference image of the MRI head image(a) is processed as an example. By searching the image, the three modes (-1,0), (1,0) and (0,0) were found as the most frequently appearing modes, they appeared 523, 520, and 520 times respectively. Since pixel values of the first difference image are from -255 to 255, mode (-1,0) was assigned as -256, mode (1,0) as -257, and mode (0,0) as -258. The size of the image was reduced by (523 + 520 + 520) pixels. In the Huffman coding, -256, -257 and -258 will be presented as symbols.

Based on the joint probability density function, the most frequently appearing mode was around (0,0). A program was developed to search those frequently appearing modes. The most frequently appearing modes and their occurrences were found. To test the coding performance, different numbers of modes were used to reduce the image volume. The bit rates resulting from using different number of modes are given in the next section.

Huffman Code

Huffman coding is a coding method based on the source symbol probabilities. The algorithm is optimum in the sense that the average number of binary digits required to represent the source symbols is minimum. In image coding, the source symbols are the pixel values of the image. For an 8-bit image, the pixel values are a set of integers from 0 to 255. For the first difference images, the possible pixel values are from -255 to 255. The Huffman coding uses the probabilities of these pixel values. The frequently appearing symbols were assigned to shorter codes and the symbols that hardly appear were assigned to longer codes.

Since the image to be coded has been shrunk from the original first difference image, the source symbols consist of two parts: a set of integers from -255 to 255 and some special symbols which represent the frequently appearing modes. These special symbols were assigned to integers which are not used in the first part. Let us use the first difference image of MRI head image (a) as an example again. By doing image mode searching, the most frequently appeared modes were obtained; some of them are shown in Table 4.5.

The most frequently appearing modes were combinations of 0, 1, -1. The histogram of the image is shown in Fig 4.10. In the histogram, there are some zero gaps. The longest zero gap was at the two ends of x axis. The searched modes were fitted in these zero gaps. After the modes were fitted in the histogram, the histogram shape was changed. Fig 4.11 shows the histogram after the modes in Table 4.5 were

Table 4.5: The most frequently occurring modes in the first difference image of MRI head image(a)

Appeared times	*First pixel*	*Second pixel*
523	-1	0
520	1	0
520	0	0
518	0	1
491	-1	1
491	1	-1
485	0	-1
473	1	1
467	-1	-1
400	-2	-1
397	0	-2
395	0	2
392	-1	2
385	2	-1

inserted.

It is observed that the peak of the histogram in Fig 4.10 was decreased and another distribution was created because of inserting modes. Since the new histogram is less concentrated, the calculated entropy is expected to be larger than the one calculated for the original histogram of the first difference image. However, the entropy directly calculated from the new histogram will be adjusted accordingly, the reduction of the total volume of the image due to the image size shrinking.

Huffman codes were created for the first difference image of MRI head image(a). Entropies and total volumes were compared. A Huffman coding program which was obtained from "Numerical Recipes in C" [16] was modified to fit the image coding

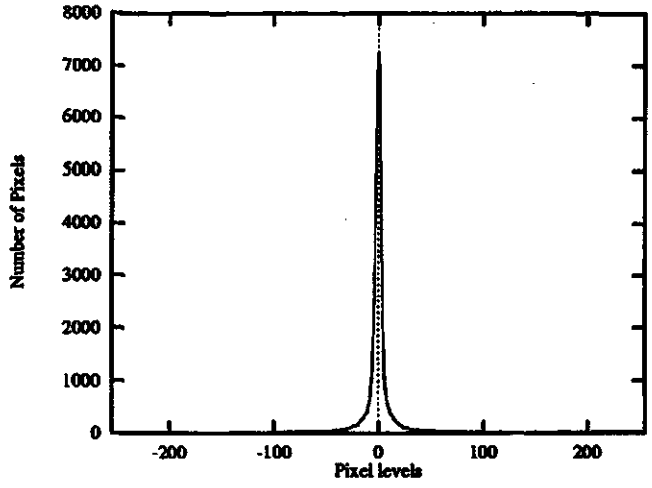


Figure 4.10: The histogram of the first difference image of MRI head image(a)

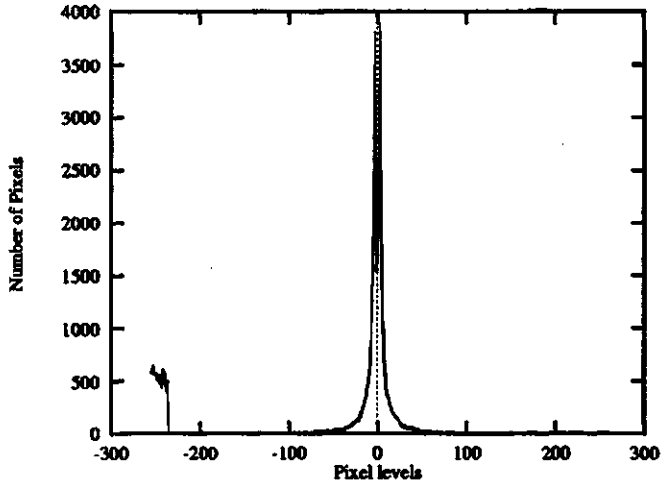


Figure 4.11: The histogram of the first difference image of MRI head image(a) after inserting the modes

need. Huffman codes were created by this program and the total image volume was calculated. The Huffman code is uniquely and instantaneously decodable. A decoding program and an image recovering program were written and tested. Experiments were conducted for several different numbers of modes.

4.4.1 Results and Analysis

Two first difference MRI head images shown in Fig 4.3 were used for the experiments. The results of inserting 0, 5, and 20 modes are shown in Table 4.6 and Table 4.7. The image shrinking with inserting 0 modes means that no mode was inserted into the first difference image, i.e. the first difference image itself was coded. Total volume was the total number of bits in the coded image. Calculated bit rate was the average bit rate based on the shrunk image size. It equals the total volume divided by the shrunk image size. Equivalent bit rate was based on the original image size. It equals the total volume divided by the original image size. As shown in the results, inserting modes did not reduce the average bit rate. Despite the fact that the image size was reduced, the total volume was increased because the shrunk image had larger entropy compared with the original image.

Fig 4.12, Fig 4.13, and Fig 4.14 show the variation of image sizes, calculated entropies, and equivalent entropies with the number of inserted modes. From these figures, it is clear that the image size is reduced with increasing the number of inserted

Table 4.6: The results of first difference image of MRI image(a)

# of inserting modes	*Image size*	*Total volume*	*Calculated bit rate(bits/pixel)*	*Equivalent bit rate(bits/pixel)*
0	65536	320312	4.887573	4.887573
5	61361	323520	5.272404	4.936523
20	54772	325456	5.942014	4.966064

Table 4.7: The results of first difference image of MRI image(b)

# of inserting modes	*Image size*	*Total volume*	*Calculated bit rate(bits/pixel)*	*Equivalent bit rate(bits/pixel)*
0	65536	340552	5.196411	5.196411
5	62575	341808	5.462373	5.215576
20	57243	343800	6.005975	5.245972

modes. However, with the increase in the number of inserted modes, the entropy of the shrunken image was increased. Consequently,the calculated bit rate was increased. As a balance of these two effects, the resulting equivalent bit rate is increased slightly.

According to the experimental results based on the MRI images, inserting modes did not improve the bit rate. The best bit rate was obtained by coding first difference images. For MRI head image (a), the bit rate by Huffman coding is 4.89 bits/pixel. The compression ratio, as defined in chapter two, was 1.64. For MRI head image (b), the bit rate after coding was 5.20 bits/pixel. The compressing gain was 1.54. Although the inserted modes method did not improve the bit rate for MRI images, it does not mean that this method can not work at all since it really depends on the histogram of a particular image. By calculating the entropies based on the histograms

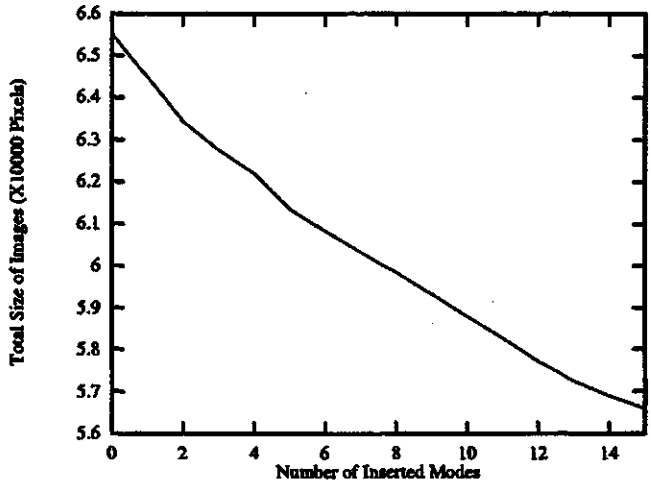


Figure 4.12: Image sizes of inserting different modes

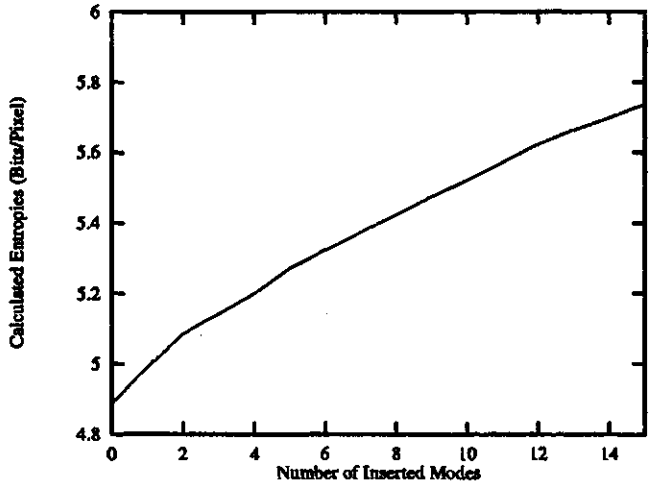


Figure 4.13: Calculated entropies of inserting different modes

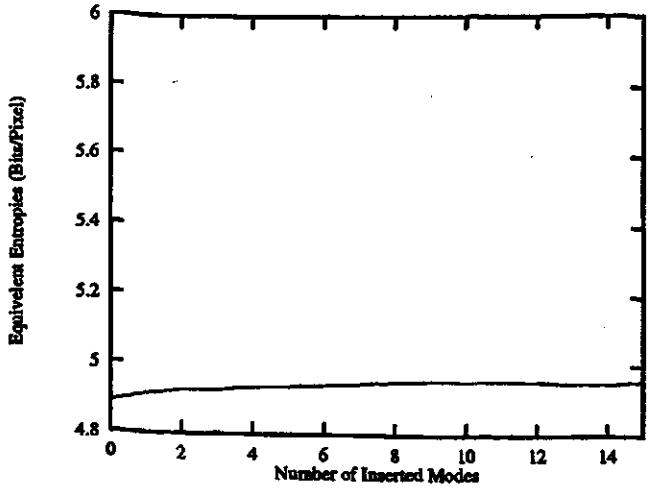


Figure 4.14: Equivalent entropies of inserting different modes

before inserting modes and after inserting modes, the minimum occurrence of modes required to improve the equivalent bit rate could be estimated. If an image data structure satisfies such conditions, the mode insertion method would improve the average bit rate.

On the other hand, lossless coding has a lower bound determined by the entropy of the image. No matter what kind of compression method is used, the bit rate after coding is greater than the entropy of the image. If a bit rate lower than the entropy is required, lossy coding has to be employed. One of the most commonly used lossy coding methods is JPEG[1]. A useful property of JPEG is that the degree of losses can be varied by adjusting compression parameters. JPEG can typically achieve 10:1 to 20:1 compression without visible loss for color images and around 5:1

[1]JPEG stands for Joint Photographic Experts Group

compression for gray-scale images. Compression of the MRI head image(a) by using JPEG was performed and some results were presented in Fig /reffig:jpeg for the sake of comparison to the lossless coding discussed. The compression ratios of the images are 27.58:1, 9.88:1, and 5.63:1, respectively.

4.5 Summary

In this chapter, two algorithms for constructing difference images were implemented and applied to the slices of MRI head images. The evaluation of these two algorithms were made by calculating entropies of difference images. According to the experimental results, the standard pixel difference coding produced less entropy than the HEDI-CS2 image, so the compression result of the standard pixel difference image is better than that of HEDI-CS2 by a small amount. But HEDI-CS2 has its own advantage; it allows an image to be viewed in a progressive manner. An advanced study for removing more redundancy was done by combining Huffman codes and the state transitions that frequently occur within a near zero range of the pixel difference scale. Although the method did not improve the bit rate for the slices of MRI head image, theoretically, it should have positive effect for some kind of images. The programs to construct difference images and modified Huffman coding were developed. Two MRI head images were used to test the algorithms and programs. For the MRI head image (256 x 256 pixels and 8 bits) head image shown in Fig 4.3 (a), the best bit rate

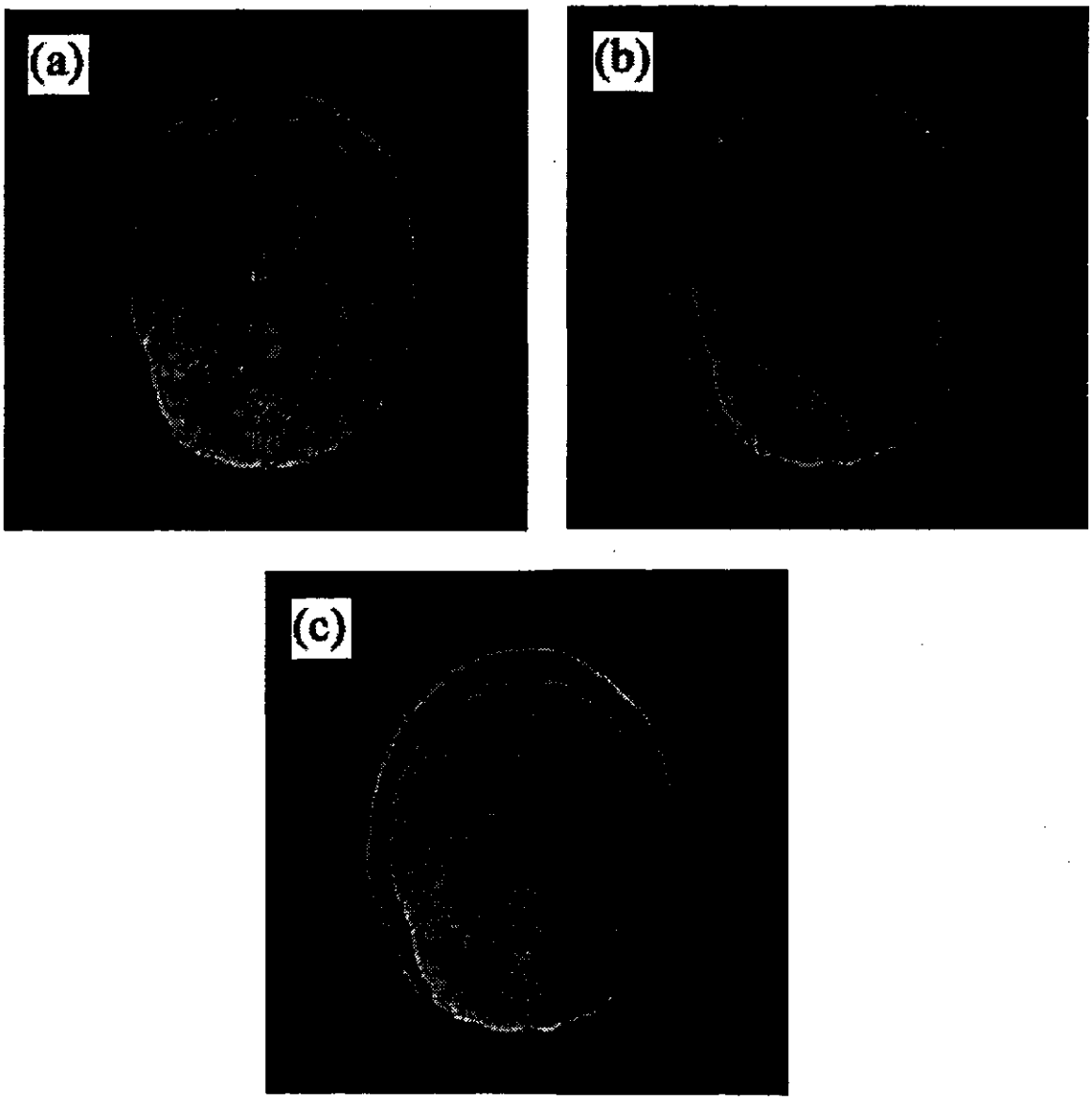


Figure 4.15: Compressed human head image by using JPEG with compression ratio (a) 27.58:1, (b) 9.88:1 and (c)5.63:1

obtained was 4.88 bits/pixel and the compression gain was 1.64.

Chapter 5

Summary and Conclusion

This thesis covers two topics related to image processing. First topic is seed classification using image analysis techniques. The second topic is implementing and evaluating two image compression methods for MRI images.

For seed classification, a machine vision system which was used to capturing the colour images of barley seeds was introduced. The colour features of the barley seeds were obtained by using an image histogram technique and an eight point border following method. A multi-layer neural network was employed as a classifier. Four barley seed varieties, Tankard, Breedn, Creme, and CDC Dolly, were used in the experiment. The feature selection is important in image analysis. As neural network inputs, colour features represented in different colour spaces, RGB, Lab, and HSI were tested. The best results were obtained by using colour features in HSI colour space.

This is consistent with KL transform theory in the sense that among colour features in these three colour spaces, HSI is the closest colour space to the KL transform.

Multi-layer neural network theory was introduced in Chapter 2. The basic element of a neural network is a neuron. The mathematical model of a neuron was given. The backpropagation training algorithm, which is an iterative gradient algorithm designed to minimize the mean squared error between the actual output and the desired output, was used to train a multi-layer neural network. Different neural structures were tested to achieve the best result. For the four varieties of barley seeds, one hundred and forty seed samples from each variety were used for neural network training and testing. The colour features of barley seeds in different colour spaces, such as HSI or RGB, were used as features to separate different varieties. Several experiments were conducted to achieve the best classification result. The results showed that using one output neuron for each variety was better than using a single output neuron for all varieties. The number of hidden layers of the neural network and the number of neurons in each hidden layer also affect the neural network performance. The best results were obtained by using a neural structure which had three inputs, two hidden layers, 20 neurons in each hidden layer, and four output neurons. The total error of the neural network was 74.19. The average root mean squared error was 0.182. The recognition accuracy of Tankard, Breedn, Creme, and CDC Dolly were 64.3%, 83.6%, 92.1%, and 90.7% respectively for individual seeds.

The future work of this research area could include several avenues. The barley seed features used in this thesis were colour features. Some other features, such as shape used in barley seed classification by Romaniuk [18], could be combined in order to obtain better classification results.

The barley seed feature extraction program and neural network program should be merged onto the image capturing platform. The image acquisition and analysis for barley seed classification could be achieved in a real time system. For constructing a real time system, in addition to the existing facilities, an automated seed feeder which can automatically put seed samples on the field of view needs to be incorporated. For a better classification result, colour should be calibrated every time before data acquisition is excuted.

The objective of image compression is to represent a digital image using as few bits as possible. Lossless coding, one of the popular image compression methods, requires that the original image should be reconstructed without losing any information. In the image compression study, two algorithms for constructing difference images were implemented and evaluated. For MRI human head images, the entropy of a standard difference image is smaller than the entropy of circle sub-sampling hierarchy embedded differential image (HEDI-CS). The programs for constructing these two kinds of images were developed and applied to human head MRI images. There is no significant difference for the compression results of these two methods. However, the

HEI-CS2 compression has an advantage of progressive transmission.

Based on the information entropy of the image data, Huffman coding is an optimal coding in the sense that it minimizes the average bit rate. To reduce the correlation of adjacent pixels, a joint probability of the difference MRI images was calculated. A modified Huffman coding were implemented in order to get better compression results. By applying the compressed Huffman coding to the human head image, the compression ratio did not improve. The lowest bit rate for a human head image in Fig 4.3 (a), which was obtained by applying Huffman coding to the first difference image, was 4.89.

The future work for this research area include the following avenues. The effect of inserting modes into a difference image for a better compression result could be investigated further: inserting modes could have different length of adjacent pixels in one or two dimensions. Images could be divided into several areas and different coding methods will be applied to different areas. Transferring images in real time from one computer to another could be investigated.

Bibliography

[1] Gregory Scott Broten. A neural network approach to analyzing multi-component mixtures. Master's thesis, University of Saskatchewan, September 1992.

[2] Kenneth R. Castleman. *Digital Image Processing.* Prentice-Hall, Inc., 1979.

[3] Keinosuke Fukunaga. *Introduction to Statistical Pattern Recognition.* Academic Press, New York, 1972.

[4] Jeffrey L. Hehn. Measurement of seed features using macintosh-based imaging. Master's thesis, University of Saskatchewan, August 1991.

[5] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci.*, 79:2554–2558, 1982.

[6] Whoi-Yul Kim, Poras T.Balsara, David T. Harper, and Jon Wong Park. Hierarchy embedded differential image for progressive transmission using lossless compression. *IEEE Transactions on Circuits and Systems for Video Technology*, 5(1):1–13, February 1995.

[7] Jae S. Lim. *Two-Dimensional Signal and Image Processing.* Prentice-Hall,Inc., New Jersey, 1990.

[8] Timothy Masters. *Practical Neural Network Recipes in C++.* Academic Press, Inc., San Diego, 1993.

[9] William S. Meisel. *Computer-oriented Approaches to Pattern Recognition.* Academic Press, Inc., New York, 1972.

[10] Byron K. Miller and Michael J. Delwiche. A color vision system for peach grading. *Transactions of the ASAE*, 32(4):1484–1490, 1989.

[11] M. R. Neuman, H. D. Sapirstein, E. Shwedyk, and W. Bushuk. Wheat grain colour analysis by digital image processing i. methodology. *Journal of Cereal Science*, 10:175–182, 1989.

[12] M. R. Neuman, H. D. Sapirstein, E. Shwedyk, and W. Bushuk. Wheat grain colour analysis by digital image processing ii. wheat class discrimination. *Journal of Cereal Science*, 10:183–188, 1989.

[13] Arun N.Netravali and Barry G.Haskell. *Digital Pictures.* Plenum Press, New York, 1988.

[14] Suranjan Panigrahi. Colour classification of corn germplasm using computer vision. *Proceedings of SPIE*, 1836:78–90, 1992.

[15] William K. Pratt. *Digital Image Processing.* John Wiley & Sons, Inc., New York, 1991.

[16] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C.* Press Syndicate of the University of Cambridge, New York, 1988.

[17] John G. Proakis. *Digital Communications.* McGraw-Hill, New York, 1989.

[18] Maurice D. Romaniuk. Pattern recognition of barley seeds using fourier descriptors and neural network. Master's thesis, University of Saskatchwan, August 1994.

[19] Azriel Rosenfeld. Connectivity in digital pictures. *Journal of the Association for Computing Machinery*, 17(1):146–160, January 1970.

[20] John C. Russ. *The Image Processing Handbook.* CRC Press, Inc., Boca Raton, Florida, 1992.

[21] C. N. Thai and R.L. Shewfelt. Modeling sensory color quality of tomato and peach: neural networks and statistical regression. *Transaction of the ASAE*, 34(3):950–955, 1991.

Appendix A Backpropagation Algorithm

As one of the most successful techniques of training multi-layer neural networks, the backpropagation algorithm was developed by *Rumelhart et al.* in 1986. A well trained multi-layer neural network has its optimal weight values which minimize the error between the desired and actual neural outputs for a given input pattern. By applying the backpropagation algorithm, weight values of a neural network are adjusted to reduce the training error.

After the inputs propagate forward through a neural network, a summed squared error is calculated from the neurons in the output layer by comparing the actual neural outputs with desired outputs. Then, the error propagates backward through the network to the input layer. During the backpropagation, an instantaneous error gradient, which indicates the change rate of error, is calculated as a square error derivative from each pair of neurons between two adjacent layers. The neural weights

are adjusted in a direction opposite to the corresponding error gradients.

When the kth set of data is presented in a neural network, the error for the ith neuron in the output layer is defined as

$$\epsilon_{ki} = d_{ki} - y_{ki} \tag{5.1}$$

where d_{ki} is the desired output and y_{ki} is the actual output. For a multi-layer neural network with N outputs, the total *summed squared error* is defined as

$$\epsilon_k^2 = \sum_{i=1}^{N} \epsilon_{ki}^2 \tag{5.2}$$

The *square error derivative* for the jth neuron in the lth layer, δ_j^l, is defined as

$$\delta_{kj}^l = -\frac{1}{2}\frac{\partial \epsilon_k^2}{\partial s_{kj}^l} \tag{5.3}$$

where s_{kj}^l is the somatic aggregate for the jth neuron in the lth layer of the neural network. Hence, the square error derivative indicates how sensitive the total summed squared error is to changes in the somatic aggregates for each neuron. The square error derivatives for the output neurons are calculated by multiplying the error of the neuron with the derivative of the associated sigmoidal nonlinearity.

The derivatives for the neurons in the hidden layers are determined by using the chain rule. For example, the square error derivative, δ_{km}^l, for the mth neuron in the lth layer, which is adjacent to the output layer, is given as

$$\begin{aligned} \delta_{km}^l &= -\frac{1}{2}\sum_{i=1}^{N}\left(\frac{\partial \epsilon_k^2}{\partial s_{ki}}\frac{\partial s_{ki}}{\partial s_{km}^l}\right) \\ &= -\frac{1}{2}\sum_{i=1}^{N}\left(\delta_{ki}\frac{\partial s_{ki}}{\partial s_{km}^l}\right) \end{aligned} \tag{5.4}$$

where s_{ki} is the somatic aggregate of the ith neuron in the output layer, s^l_{km} the somatic aggregate of the mth neuron in the lth layer and δ_{ki} the square error derivative of the ith neuron in the output layer. $\partial s_{ki}/\partial s^l_{km}$ can be represented as the weight between the ith neuron in the output layer and the mth neuron in the lth layer times the sigmoidal nonlinearity associated with the mth neuron in the lth layer.

After determining the square error derivatives for all neurons in the neural network, the neuron weights can be adjusted. In the backpropagation algorithm, the instantaneous error gradient, Δ_k, is defined as

$$\Delta_k = \frac{\partial \epsilon_k^2}{\partial W_k} \tag{5.5}$$

where W_k is the weights of the neural network. For an input vector, X_k, the output S_k is represented as

$$S_k = W_k X_k \tag{5.6}$$

By using the chain rule, the gradient can be rewritten as

$$\begin{aligned} \Delta_k &= \frac{\partial \epsilon_k^2}{\partial S_k}\frac{\partial S_k}{\partial W_k} \\ &= \frac{\partial \epsilon_k^2}{\partial S_k}\frac{\partial W_k X_k}{\partial W_k} \\ &= \frac{\partial \epsilon_k^2}{\partial S_k} X_k \end{aligned} \tag{5.7}$$

Substituting Equation 5.3 into 5.7 gives

$$\Delta_k = -2\delta_k X_k \tag{5.8}$$

According to the backpropagation algorithm, the weights are adjusted in a direction opposite to the error gradient. The adjusted weights are given as

$$\begin{aligned} W_{k+1} &= W_k + \mu(-\Delta_k) \\ &= W_k + 2\mu\delta_k X_k \end{aligned} \tag{5.9}$$

where μ is the learning rate, which should be a positive number between 0.1 and 0.3. The learning process can be accelerated using a proper learning rate.

The amount of the weight adjustment from the previous weights is determined by the neuron inputs scaled by the associated square error derivative and the learning rate. The weight adjustment process is also called the training process of a neural network. A well trained neural network can be achieved by repeating the training process.