INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Appearance Based Object Recognition Using Independent Component Analysis

Harkirat S. Sahambi

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Applied Science at

Concordia University

Montréal, Québec, Canada

September 2000

ABSTRACT

Appearance Based Object Recognition Using
Independent Component Analysis

Harkirat S. Sahambi

In recent years there has been a renewed interest in appearance based 3-dimensional object recognition. It involves the matching of appearance of the given image with the appearance of the images in a database. All possible views of an object define its workspace, known as visual workspace. This workspace is coarsely sampled and projected onto a lower-dimensional space and represented as workspace manifold. The dimensionality reduction is usually done using Karhunen-Loeve transform (KLT), or principal component analysis (PCA). The lower dimensional space has been called the eigenspace. For recognition, the test image is projected likewise onto the eigenspace and its position on the appearance manifold is used for the recognition phase.

Although PCA has been commonly used for feature extraction and data compression, it takes only second order statistics into account. In object recognition, there are many situations in which features based on only second order statistics are not sufficient. To take into account higher order statistics, Independent Component Analysis(ICA) has been proposed. ICA has been used to reduce the dimensionality of the data and to project the features on a lower dimensional space using Hebbian-like learning rules in unsupervised learning. Furthermore, some of the ICA basis vectors correspond to wavelet type filters that are sensitive to local features

and spatial frequencies in the images and still others correspond to edge detectors. In comparison, in PCA the basis vectors correspond to filters sensitive to spatial frequencies. This makes ICA superior to PCA in extracting useful information or features from the images in a completely unsupervised manner.

This thesis presents results on appearance based 3-dimensional object recognition accomplished by using ICA. Two database of images captured by a CCD camera were used. The workspace was then sampled in a certain manner. Features were extracted from the sampled images using ICA employing information maximization approach. The features of all the objects thus obtained were saved in a database which formed the workspace manifold. The test images was also represented in a similar manner. Recognition was then performed by locating the closest point in the manifold which gave the identity and view (or pose) of the object. ICA did perform better than PCA in one of the databases, but surprisingly, it's performance was no better than PCA in the case of the second database. Thus suggesting that the use of ICA may not give better results than PCA in general and that the application of ICA is highly data dependent. Various factors affecting the difference in the recognition performance using both the methods are also discussed.

Dedicated to my parents ....

ACKNOWLEDGEMENTS

Many people have helped me, directly and indirectly, in completing my masters thesis. First and foremost, I would like to extend my sincere thanks to my advisor, Dr. K. Khorasani. His help has been invaluable in my graduate studies. The discussions he had with me relating to my work played the most significant role in the writing of this thesis. I appreciate the time he devoted to help me understand the concepts involved and to make me better appreciate the thesis work.

It would have been tremendously difficult for me to successfully complete my thesis without the unfading support of my family members and friends. I thank my parents for their unwavering confidence in me and their constant encouragement. I also thank my brothers, Kamal Deep and Dr. J. S. Sahambi for all kinds of help, technical as well as non-technical, that they extended towards my graduate studies. I would also like to thank my friends, Sanjeev, Prasad, Pondugala and Bhanu who helped me remove the numerous mistakes that I never seemed to notice while reading the proofs of my thesis chapters.

Also, special thanks to our system administrators, Daniel, Wojciech and Claude, in helping me utilize the power of Unix and shell scripts to deal with hundreds of image files. I also thank Guy and Silviu for wiring up the CCD camera and the frame grabber and for making it work. The Inter Library Loans people also deserve my thanks for helping me locate papers not available in local libraries.

And thanks to all those people who have not been mentioned here but who have helped me during my masters, however small that help may have been.

Harkirat S. Sahambi

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

A major thrust area in present day research in intelligent systems is to make better, smarter, efficient, and autonomous machines. Machines that can not only walk, talk, see, but can also feel, think, and take decisions on their own. In other words the machines that can not only perform the tasks that humans usually do, but can also perform them with higher accuracy and precision. Computers are a common example, they are faster and more precise at numerical computations than humans. Then there are sensors that can measure temperatures, pressure, flows, speed, etc., things that humans cannot do with any reasonable accuracy. Consider memory devices available in the market in which all the volumes of a typical encyclopedia can be stored, something the human memory completely falls short of. These are just a few examples to show that there are machines that outperform humans by a big margin. Still, humans are superior to them.

Humans have brain. It is arguably the most complex machine known to man today. It may not be precise, but it is intelligent. It learns from experience, makes decisions based on learning, can recall long forgotten memories depending on the situation that triggered the recall mechanism, and some also think that it has powers beyond the ones that we know about and understand today, e.g. extra sensory perception, telekinesis, ability to make intelligent *hunches*, gut feeling, etc.

All engineers will agree that human body, controlled by the brain, is a marvelous feat of engineering given the natural circumstances in which the humans have evolved. And they will also agree that making a machine as intelligent and as capable as the brain with present day's technology, and knowledge, is not currently feasible. It took thousands of years for the humans to evolve to what they are today. It would be naive to think that a brain-like machine can be built within a few decades. But then these very difficulties are food for thought for the engineers and scientists. For the humans have an innate desire to learn more; they continually seek challenges to overcome. Where the task of making a brain-like machine is concerned, engineers have found that given the present theoretical and practical know-how, it is more practical to work making machines that perform only a subset of operations that the brain does. One of these operations being that of vision.

It is important to get a feel of what the human vision system, as we know it, actually does. Sight is a faculty that humans usually take for granted and seldom ponder about it. Human visual system is a complex neuro-physiological system composed of optical lenses, sensors, communication channels, encoder-decoders, memory and decision-making computers. It may be safely stated, that the knowledge to understand it has to come from a wide range of fields of sciences and engineering, e.g. electrical engineering, physiology, biology, chemistry, communication theory, etc. Humans can see and perceive the world around them right from birth. As they grow old, they can also recall and recognize people and objects around them. Sight, perception, recall, and recognition are the tasks performed by humans with no apparent difficulty, yet tasks which are extremely difficult to duplicate in an artificial machine. The research to mimic the human vision systems is a continuing research topic. New theories continue to be introduced, old theories continue to be improved upon, yet we are far from making a machine that can be said to be even reasonably similar to the human vision system. Object recognition is one area which is continuing to pose a formidable challenge to the researches.

The work in this thesis is concentrated towards the view based three dimensional (3-D) object recognition. It involves the representation of the images of objects in such a way as to make the recognition task more efficient. To fully appreciate the motivation of this work and the connection of the approach followed to the human brain, it will be useful to first introduce the human vision system and to show how it transmits the data available through eyes to the brains and how the data is compressed to be used later for recall and recognition.

1.1 The human vision system

In this section the human vision system is briefly introduced. The reader is referred to [1, 2, 3, 4, 5] for further details.

Human vision is a very complex neuro-physiological process. The light reflected from the surrounding world passes through the lenses of the eye and is projected onto the retina. The retina includes a receptor layer at the back and the ganglion cell layer at the top. Response to the visual stimuli in the retina starts in the photo-receptors in receptive layer and travels through other layers to the ganglion cells in the ganglion cell layer. The receptive layer consists of two types of photo receptors: rods (for luminance) and cones (for chrominance). The human receptor layer consists of approximately 120 million rods and 6 million cones. These photo-receptors generate impulses when light strikes them, and these impulses are passed to the ganglion cells. Ganglion cells groups together to make receptive fields, with a number of ganglion cells belonging to one such field. The receptive fields are sensitive to various kinds of visual stimuli - on-center/off-surround, off-center/on-surround, etc. The part of the human eye responsible for most sensitive and acute vision is the Fovea, directly behind the lens. It is about 0.2 mm in diameter and only cones are present in it. All these neural components - retina, fovea, rods, and cones - are the starting point of the sensing and encoding of the visual stimuli. After

the visual information has been coded, it is transmitted to the brain.

Optic disk is the portion of the eyeball from where the axons, the neural transmission lines, leave the eyeball towards the brain through the optic nerve. The optic nerve may be considered as a transmission channel that carries the information, in the form of action potentials. Since the area of the eyeball where the optic disk is located is used for the passage of the axons to the optic nerve, it has no photo-receptors and is called the blind spot. The information is carried along the optic nerve, a major relay station known as the Lateral Geniculate Nucleus (LGN). At LGN, the information from both the eyes is separated - left visual field goes to the right hemisphere of the brain and the right visual field goes to the left hemisphere of the brain. The visual information is then carried from the LGN to the primary visual cortex (at the back of the human cortex). At the LGN, the visual information is further divided into three categories:

1. movement and the overall features of the stimulus
2. detailed information about the stimulus
3. color.

This shows that some kind of analysis is done on the information for the subdivision. The signal then travels to selected areas of the primary visual cortex, or Area V1, which curves around a deep fissure at the back of the brain. Hubel and Wiesel described the V1 area as a highly structured arrangement of cell selectivity [6, 7]. Area V1 has striking regularity in the organization of the cells which deal with selective visual stimuli, e.g. some may be sensitive to bars of lights and others may be sensitive to abrupt changes of light. In other words, area V1 comprises of feature detectors - edge, bar, motion, orientation, binocularity, etc. The group of cells in layer 2 and layer 3 of area V1 project the information onto the higher areas of the brain, such as secondary visual cortex or the striate cortex. Then an explosion of connections occurs to further higher areas of the brain. These include

- Ventral Pathway - This leads to the temporal lobe. It is also called the "*what* path" and plays an important role in object recognition and visual attention.
- Dorsal Pathway - This leads to the parietal lobe. It is also called the "*where* pathway" and plays an important role in spatial aspects of the visual information (motion and position) and visiomotor control, or eye movement.

It is thought these subdivisions are done to segregate the information into different categories. This region, and beyond, is the place in the brain where image *perception* occurs. This portion of the brain is where the objects are recognized and associated with past experiences.

The pattern should be clear now. The visual information is processed right from the beginning. The eye smoothens the image and can be thought of as having a low pass filter hardwired within itself. Then low-level image processing - edge , motion, orientation detection, etc. - occurs further down the visual pathway. And as the information reaches the higher levels of the brain, it becomes more abstract and global and is used to make higher level decisions by the brain. This, is the human visual system. Although a lot is yet to be known about it, but from whatever is already known, it can be easily seen that it is a massively parallel neural image processing and a decision making system. Mimicking this with artificial machines and computers is no easy task. In this thesis, the focus is on the image representation for the purpose of view-based 3-D object recognition.

1.2 Past work in view-based 3-D object recognition

It is believed that the human brain performs object recognition in the following steps:

1. It takes into account the brightness of the scene. This is done in 2-D.

2. It then considers the depth in the object under consideration. This is done to incorporate, in a sense, the 3-D structure of the object. This is called $2\frac{2}{1}$ vision

3. It combines the data from the above two steps and stores it in memory as a model and/or tries to match it with models already stored in its memory.

These steps are believed to be aimed at making a view-invariant representation of the object to facilitate 3-D recognition.

In computer vision the recognition system typically involves some sort of a sensor, the use of a model database in which all the objects' models' representations are saved, and a decision making ability. When an object is viewed by a sensor, e.g. a CCD camera, the digitized image is processed so as to represent it in the same way as the models are represented in the database. Then a recognition algorithm tries to find the model to which the object best matches. A schematic of a typical system is shown in Figure 1.1. For the view-based recognition, the representations take

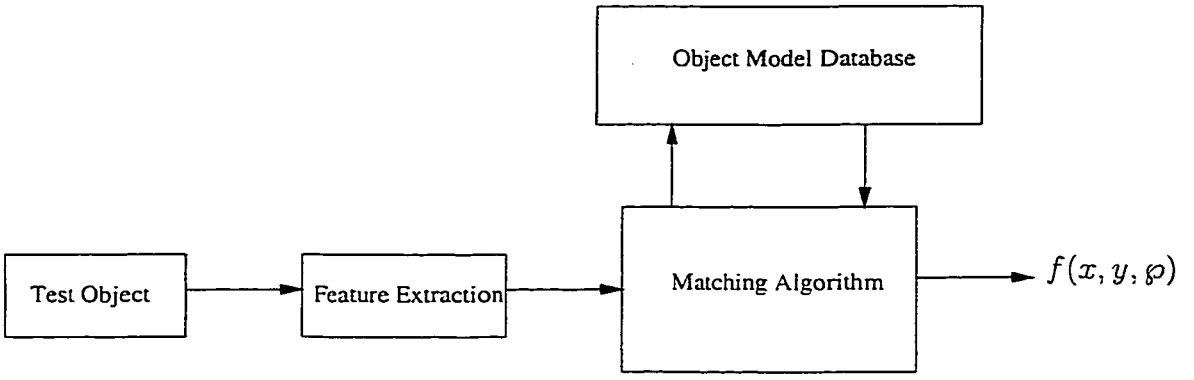


Figure 1.1: Block diagram of a typical object recognition system. In three dimensional object recognition, the pose of the object is also considered. The result, $f(x, y, \wp)$, is the best matching candidate in the model database and it depends on the spatial properties of the image and its pose, $\wp$.

into account the appearance of the object. One of the most useful application of a machine vision system is in robots. The visual stimuli is a very useful information in such systems. To make robots see, the need to recognize objects is a must. Traditionally robots vision systems have utilized the shape of objects as a cue for recognition [8, 9]. The main drawback of shape based systems is that they have implicit limitations in acquiring new models for the database. The database needs to be built by using explicit models and the robot cannot do it on its own. In appearance based recognition a new image of object can be easily considered as a novel object and added to the database.

To achieve 3-D object recognition, the pose of the objects are also saved in the database. Then the objective of the algorithm is not only to recognize the object correctly but also to identify its pose as viewed by the camera (Figure 1.1).

1.2.1 What is 3-D "appearance based"object recognition?

Many approaches have been introduced for 3-D object recognition (3DOR). The conventional, and the usual approaches involve the use of edge information in one form or another. Edges contain a great amount of information in a scene. It is also known that the eye pays more attention to edges in an image[1]. This has prompted the use of edge information in the development of 3-D object recognition algorithms.Various methods ranging from parametric bicubic patches [10] and super-quadratics [11] to the use of polynomials of higher than 2 degrees for three dimensional (3-D) objects [12, 13] have been proposed in the literature. Traditionally robot vision systems have also utilized shape of objects for recognition [8, 9]. Many of these methods explicitly exploit the features extracted from only the shape of the objects (i.e. lines, curves and vertices, also called *geometric features*) [14, 15] and

[1]This could be the reason why when two persons meet they look towards the eyes of the other person. In the human face, the region around the eyes has most edge information which tends to attract other people's attention.

do not take into account other factors that also contribute to the image representation, namely reflectance and illumination; the factors that do not affect intrinsic shape of an object but do significantly determine how the object is *viewed*. The approaches that take explicitly these factors into account for use in object recognition have been categorized as view based or appearance based object recognition methods [16, 17, 18, 19, 20, 21].

Thus in contrast to the geometric features, appearance of an object is the combined effect of its shape, reflectance properties, pose, and the illumination. As the database takes into account all of these attributes, appearance based recognition tends to take into account the combined effect of all the above factors. Also, the representations can be obtained through an automatic learning procedure and do not require the explicit specification of object models.

1.2.2 Earlier work in view-based 3D object recognition

The central idea in the appearance based approach is to represent the images in terms of their projection into a relatively low-dimensional space which captures the important characteristics of the object to be recognized. One popular method to obtain a low-dimensional space is the Karhunen-Loéve transform (KLT). The use of KLT for feature extraction from face images was first suggested by Sirovich and Kirby [22]. This method was further developed by Turk and Pentland [18, 19, 23] to give a particularly successful application of face recognition. In this application, the face images are aligned according to the position of the main features in the face image - the location of eyes, nose, lips, etc. The face images are scaled to be approximately of the same size. The face database is then represented onto a low-dimensional space using KLT, a statistical method which is called principal component analysis [24, 25]. The low-dimensional space has been called the *eigenspace*, and the projections of the images onto this space have been called the *eigenimages*, and *eigenfaces* in the case of face images.

Although face recognition applications dealt with only 2-D object recognition, their successful application prompted the extension of the method to 3-D object recognition. In three dimensions, pose becomes another parameter in addition to illumination, reflectance, etc. So the vision system also needs to memorize the appearance of the object from different pose angles (from around the object). A substantial amount of work has been done along these lines in the last few years [16, 21, 26, 27, 17, 20]. In these methods, the images of the objects are taken from various pose angles, and their compressed features are saved in a database. The images consist of the object in the image with an uncluttered background. In almost all the cases there is no occlusion of the objects. The test images are taken in similar conditions and the features extracted from them are matched against the database to determine the identity of the object along with its pose angle. Occlusion has been a difficult problem in these methods. But then it can be thought of as not being a problem but something that cannot be avoided. After all, in many situations the human vision system also fails to recognize objects when they are occluded by other objects. At the same time, humans can and do recognize occluded objects. Recently, attempts have been made to take occlusion into account in appearance based recognition (refer to [28] for details).

1.3 Dimensionality reduction and image representation

In computer vision, the size of the data is usually very large. For the purpose of object recognition, the model database has to be built. If the images are saved in their raw form, then it not only requires prohibitively huge amount of memory just to store the database, but it also makes it even more difficult to match the test object to the model images. A typical image can be of 100×100 pixels which can be represented as a $10,000$ dimensional column vector. It can be clearly seen

that the dimension of the data is a serious problem. This necessitates the need to represent the data in a lower dimension. This is usually done by projecting the data on to a lower-dimensional space. This process is called image compression in image processing community and feature extraction in computer vision and pattern recognition communities.

In the case of recognition, the process of feature extraction involves selecting and storing those features that possess the most crucial information related to the recognition. Rest of the features are discarded. The nature and the number of features are usually decided by the problem at hand, e.g. they could be different for face recognition as compared to those needed for vehicle recognition. Thus a major problem is how to represent the images so that only the useful features are obtained and saved. Usually, this involves a transformation of the image, which lies in the *image space*, to its features, which lie in the *feature space*. Thus the problem of representation boils down to finding the best transform that gives the required recognition performance.

There are many transformations that are used in image processing, e.g. the Fourier transform, Gabor transform, wavelets, etc. The aim of all these is to represent the data in some sort of basic building blocks, called the *basis functions*. In appearance based recognition, usually KLT has been used for feature extraction. It projects the image on to a new set of orthonormal axes (also called eigenimages). When these are ranked in decreasing order of significance, the amount of information decreases with the rank of the axes. Dimensionality reduction is achieved by keeping only the first few axes and by discarding the rest, thus insuring that only the least significant information is discarded. But this raises the question "Is the KLT the best possible transformation for recognition purposes?" It is the optimal transform as far as data compression and reconstruction is concerned. But the requirements of a recognition system could be different from a reconstruction system. In fact, a recognition system need not reconstruct data at all, all that is needed

is that the object of interest be accurately identified or recognized. In the human visual system also, the image is not transmitted as it is from the eye to the visual cortex. Its encoding starts right from the rods and cones, where processing noise also gets added, and only abstract representation of the image reaches the visual cortex along with the post processing noise after the optic nerve. This is explained in the block diagram of Figure 1.2.

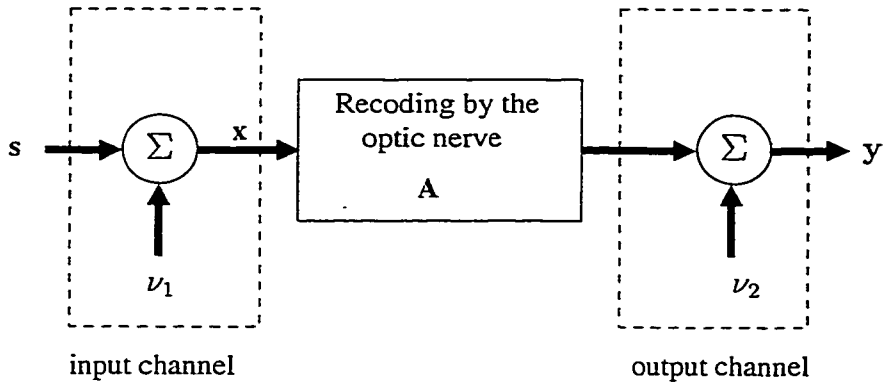


Figure 1.2: Block diagram of the human perceptual system. The scene, s, is cluttered with noise, ν_1, at input stage. The signal, s, is then recoded by the optics nerve channel, described by matrix A, to give y at the output with the post-encoding noise ν_2 [29, 30].

The way in which the information is transmitted from the eye to the higher levels of reasoning in the brain and represented efficiently is a continuing research topic. The motivation is that through millions of years of evolution the human visual system has learned efficient coding strategies for representing natural images. Information theory has long been considered as a tool to explain this information transfer from the eye to the brain and its representation [31]. This approach is called the information-theoretic approach to explain the efficient perceptual processing. It is believed that efficiency in mammalian visual system is achieved by transforming

the input in such a manner that reduces the redundancy due to complex statistical dependencies among the elements of the input signal. In the literature, this is commonly referred to as Barlow's principal of redundancy reduction [32, 33, 34]. The goal of the visual system is believed to extract the statistical dependencies from the visual information and to represent it in terms of a collection of independent events. Using this strategy, successful work has been done towards understanding the response strategies of the retinal ganglion cells [29, 35, 36, 37, 38]. But these works considered redundancy due to linear pairwise correlations among image pixels, in other words they considered first and second order statistics only. But images have information of order higher than two, e.g. in curves, fractal-like edges, etc. [39, 40]. Thus, higher order statistics need to be taken into account to reduce the redundancy from the natural scenes to achieve better representation.

Another way to explain the above approach is to consider the images as being made of independent basis functions. In other words, the bases functions are the building blocks of the scenes. The functions are weighed and summed to make the scene that is perceived by the eye. This can be written in matrix form as

$$\mathbf{x} = \Phi\mathbf{b} \tag{1.1}$$

where $\mathbf{x}$ is the seen image, the columns of Φ are called the basis functions in general and the column vector $\mathbf{b}$ has weights as its rows. This formula is known as image synthesis formula.

The visual system then performs feature extraction of various levels as the image signal travels along the optic nerve to the visual cortex. In other words, the visual system breaks up the image to achieve an efficient representation in the brain, which can be written as

$$\mathbf{a} = \Psi^T\mathbf{x} \tag{1.2}$$

where the columns of the matrix Ψ are the spatial weighting functions, and the columns of $\mathbf{a}$ are the representations. This relationship is also known as image

analysis. Thus, if the representations are independent of one another, they could be considered to be similar to the ones that the human visual system uses.

1.4 Unsupervised Neural Networks

Since the work done in this thesis uses unsupervised neural networks, these are briefly introduced here. Further information about neural network training can be found in [30, 41, 42].

Neural networks can be broadly classified into two categories:

1. Supervised neural networks, and,
2. Unsupervised neural networks.

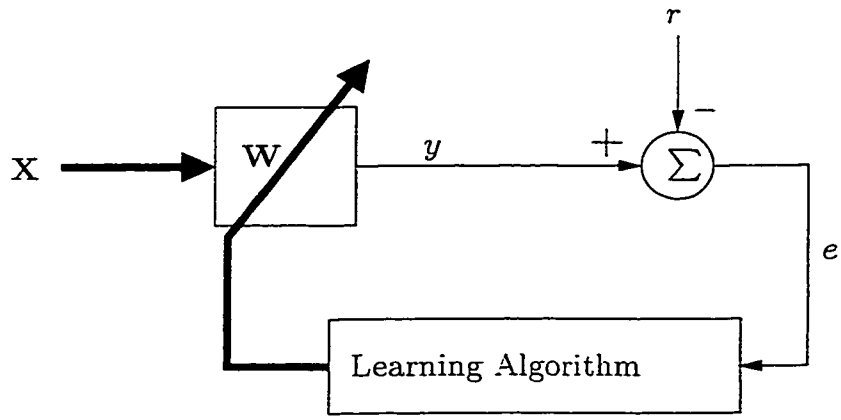


Figure 1.3: Block diagram of a supervised multi-input single-output neural network. The thick lines show the path of multidimensional signals, and the thin lines show the path of a scalar signal. The error signal, e, is used by the adaptation algorithm to update the weights in the learning process. In most of the cases, the target is to minimize the error.

In supervised neural networks, the training proceeds in the presence of a

reference signal, also called a teacher. Block diagram of a multi-input single-output neural network is shown in Figure 1.3. The input, x, is weighted by the weight matrix, w, and summed to give the output, y. The training consists of comparing the output with a given reference signal, r, and by changing the weight matrix by some adaptation rule, which is usually a cost function depending on the input/output and the reference signal.

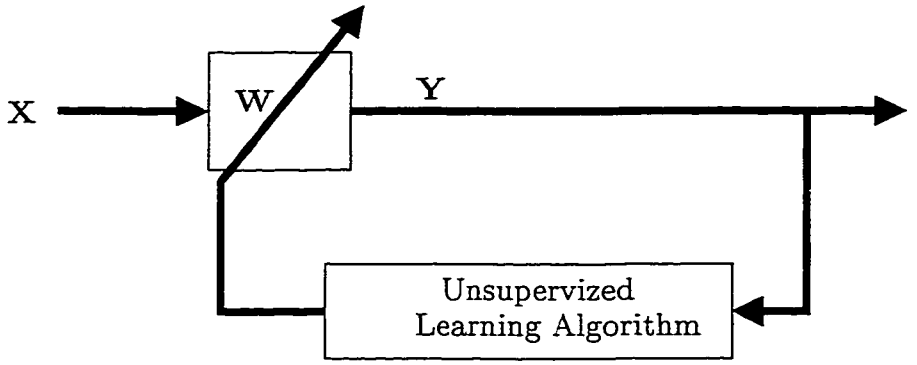


Figure 1.4: Block diagram of an unsupervised multi-input multi-output neural network. The thick lines show the path of multidimensional signals.

In unsupervised neural networks, there is no reference signal. In other words, the neural network tries to learn without a teacher. This is shown in a block diagram in Figure 1.4. Since there is no teacher for training, the cost function usually depends on the nature of information at the output. Such kind of neural networks are useful to extract information, or to search for patterns, that are not known *a priori*. This makes unsupervised networks more attractive, as the network need not be told what to look for, rather it tries to find the useful information hidden in the input. Exactly what kind of information it discovers and gives at the output depends

on the adaptation rule being used to update the weights. The unsupervised neural network for independent component analysis is explained in detail in Chapter 3.

1.5 Goal of this thesis

Recently, independent component analysis (ICA) has been proposed to extract independent components from observed data. It has been applied to EEG signals and for face recognition with encouraging results. To the best of the author's knowledge, ICA has not yet been applied to appearance based 3-D object recognition.

Principal components analysis gives the components of the images by taking into account only the first and second order characteristics. ICA, on the other hand, takes even higher order statistics into account. In this thesis, the application of ICA is examined to determine whether it outperforms the PCA in the recognition task. This appears to be the first work of its kind to link ICA and appearance based 3-D object recognition.

1.6 Layout of the thesis

The present chapter provided an introduction to the concept of machine vision, the mammalian vision system, and the motivation to mimic it with artificial machines. It also highlighted the formidable challenges it poses to the researchers. It was also discussed that representation of the images and objects is a significant step in the whole process of 3-D object recognition, and how it is connected with the principal of redundancy of information in natural scenes.

In Chapter 2, principal component analysis is presented, since it can be considered as a precursor to ICA. In Chapter 3, the theory behind ICA is introduced. The experimental results and our conclusions are provided in Chapter 4.

Chapter 2

Feature extraction using Principal Component Analysis (PCA)

2.1 Introduction

In this chapter, feature extraction using principal component analysis (PCA)is discussed. Using PCA, features based on second order characteristics are extracted from the given data, which in 3-D object recognition is a database of all the models with images of each model from various pose angles. PCA not only helps to extract features but it also helps in significant dimensionality reduction. This is based on the assumption that the visual data also contains a significant amount of noise in it, and also that the number of discriminating features sufficient for recognition are small in number. Thus the features which correspond to the noise or those that correspond to the non-discriminating ones can be discarded without any significant degradation to the recognition performance.

First the theory behind PCA is presented. Then unsupervised neural networks are presented that are used to learn second order structure in the data presented to them. These are networks particularly important because they do not need supervision and also because they can be used to extract any number of features.

2.2 Conventional Principal Component Analysis

In this section principal component analysis is used to represent a given image in its eigenspace.

Let an image, having L pixels in all, be represented by the vector

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_L \end{bmatrix}, \tag{2.1}$$

where x_i represents the value (real) of the i-th pixel and the column vector is formed by concatenating all the rows of the image into a row matrix and then by transposing it, resulting in an L-dimensional vector $\mathbf{x}$, $\mathbf{x}$ i.e. $\in \mathbf{R}^L$.

Let there by N total images, all L pixels long, in a database $\mathbf{X}$. The complete set of the images can then be represented in a matrix form as

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_N \end{bmatrix} \tag{2.2}$$

making $\mathbf{X}$ an $L \times N$ dimensional matrix.

The data in $\mathbf{X}$ is represented in a natural orthonormal basis. In principal component analysis, the basis is changed so that the eigenvectors of the covariance matrix of the input data become the new axes on which the data is projected [30, 43, 18, 44]. The columns of $\mathbf{X}$, i.e. all the images, are considered as observations and the rows of $\mathbf{X}$, i.e. pixel values of the images, are considered as measurements. Thus, statistically, there are L measurements for each of the N observations.

The first step in PCA method involves making the data zero mean. This is done by removing the row means from the data matrix $\mathbf{X}$. The mean of $\mathbf{X}$ is given

by

$$\mu_{\mathbf{x}} = \begin{bmatrix} \mu_{\mathbf{x}_1} \\ \mu_{\mathbf{x}_2} \\ \vdots \\ \mu_{\mathbf{x}_L} \end{bmatrix}, \tag{2.3}$$

where

$$\mu_{\mathbf{x}_j} = \frac{1}{N}\sum_{i=1}^{N} x_{ji}; \quad j = 1, 2, \cdots, L. \tag{2.4}$$

The zero mean data matrix is now given by

$$\mathbf{X}_0 = \begin{bmatrix} \mathbf{x}_1 - \mu_{\mathbf{x}} & \mathbf{x}_2 - \mu_{\mathbf{x}} & \cdots & \mathbf{x}_M - \mu_{\mathbf{x}} \end{bmatrix} \tag{2.5}$$

The covariance matrix of this zero mean data is given by

$$\mathbf{C} = E\{\mathbf{X}_0\mathbf{X}_0{}^T\} \tag{2.6}$$

$$= \frac{1}{N}\mathbf{X}_0\mathbf{X}_0{}^T \tag{2.7}$$

where $E\{\cdot\}$ denotes the expectation operator. The principal components are found by solving the following well known eigenvalue problem

$$\mathbf{CQ} = \mathbf{Q}\Lambda \tag{2.8}$$

where the columns of $\mathbf{Q}$ are the eigenvectors, denoted by $\mathbf{q}_i$, and Λ is a diagonal matrix with the eigenvalues of $\mathbf{C}$, denoted by λ_i, on its diagonal, such that

$$\mathbf{C}\mathbf{q}_i = \lambda_i\mathbf{q}_i, \quad i = 1, 2, \cdots, L \tag{2.9}$$

If the columns of $\mathbf{C}$ and Λ are rearranged so that the eigenvalues on the diagonal of Λ appear in decreasing order from the largest in magnitude to the smallest, then the eigenvector corresponding to the first eigenvalue will point towards the direction of maximum variance in the data, the next eigenvector will point towards the direction

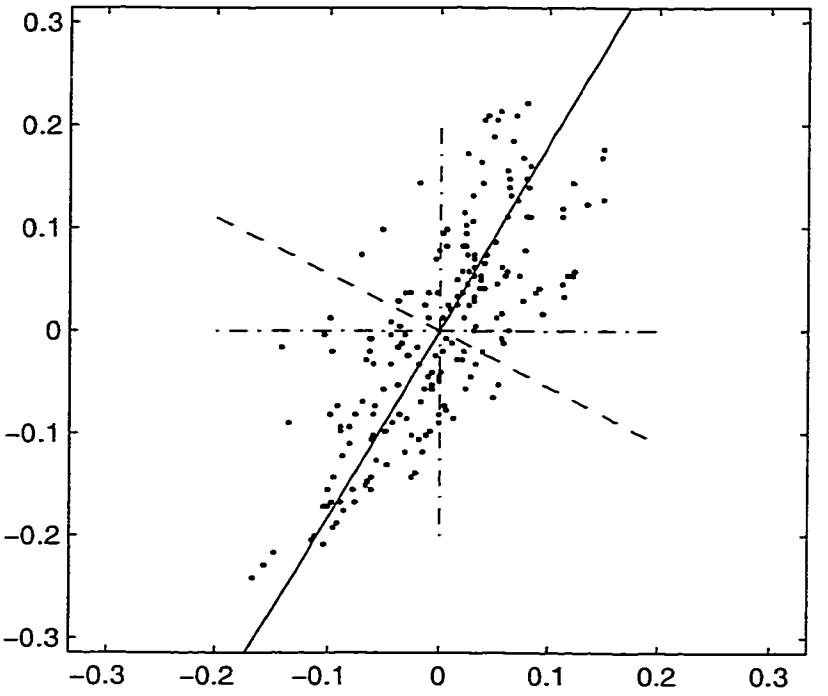


Figure 2.1: The plot of eigenvectors of a two dimensional zero mean data found by PCA. The solid line is the eigenvector corresponding to the largest eigenvalue and the dashed line shows the eigenvector corresponding to the second eigenvalue. The natural axes are shown with the dash-dot lines.

of the next highest variance, and so on, with the last eigenvector pointing towards the direction of the least variance in the data. This is illustrated in Figure 2.1 in which PCA has been performed on a zero mean data. Also, $\mathbf{C}$ is an orthogonal matrix and its columns satisfy the following relation

$$\mathbf{q}_i{}^T\mathbf{q}_j = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \tag{2.10}$$

or

$$\mathbf{Q}^T\mathbf{Q} = \mathbf{I} \tag{2.11}$$

From equations (2.8) and (2.11), finding the eigenvectors of $\mathbf{C}$ can also be considered as its diagonalization given by the following relation

$$\mathbf{Q}^T\mathbf{C}\mathbf{Q} = \mathbf{\Lambda} \tag{2.12}$$

This equation is obtained by pre-multiplying equation (2.8) with $\mathbf{Q}^T$.

Having found the eigenvectors of the covariance matrix, or the principal components of the data, the next step is to present the data in terms of this new set of basis. Note that the eigenvectors form a new orthonormal basis. The data is then represented in terms of this new basis by the data's projections, $\mathbf{\Gamma}$, onto the new set of axes, such that

$$\mathbf{\Gamma} = \mathbf{Q}^T\mathbf{X} \tag{2.13}$$

In this equation, the *representations* of the images in columns of $\mathbf{X}$ are in the corresponding columns of $\mathbf{\Gamma}$, i.e.

$$\mathbf{\Gamma} = \begin{bmatrix} \gamma_1 & \gamma_1 & \cdots & \gamma_N \end{bmatrix} \tag{2.14}$$

where γ_j is the representation of image $\mathbf{x}_j$ in the eigenspace. This can also be written as

$$\gamma_j = \mathbf{Q}^T\mathbf{x}_j, \quad j = 1, 2, \cdots, N \tag{2.15}$$

Also, it can be seen from equation (2.13) that given only the representations, γ_j, and the principal components, $\mathbf{Q}$, the original data can be reconstructed from

$$\mathbf{X} = \mathbf{Q}\mathbf{\Gamma} \tag{2.16}$$

Note that the size of $\mathbf{\Gamma}$ is the same as that of $\mathbf{X}$. The representations of the images are also known as the *eigenfeatures*.

In object recognition, the features of a test object, $\mathbf{x}_t$, are obtained by projecting the image onto the orthonormal axes of the database, i.e.

$$\gamma_t = \mathbf{Q}^T\mathbf{x}_t \tag{2.17}$$

and recognition is achieved by finding the best match in Γ to γ_t. Thus, to identify an object from some objects in a given database, the test object's representation is compared with the representations of all the objects in the database. It can be clearly seen that this can be computationally very intensive. But it has been found that a complete representation is not necessary for a reasonably successful recognition. The representation length is reduced by keeping only the first first few principal components, those accounting for the maximum variance in the data, and by discarding the rest. This can also be seen from the energy point of view. Most of the energy of the data is concentrated only in the first few principal components, and the rest do not carry any significant information, in fact, most of it can usually be considered as noise. This reduction in the size of representations is called image compression in image processing and feature extraction in pattern recognition. Let only the first n, out of a total of N representations be retained. This can be written as

$$\Gamma_n = \mathbf{Q}_n^T \mathbf{X} \tag{2.18}$$

where Γ_n is an $n \times L$ matrix, and

$$\mathbf{Q}_n^T = \begin{bmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_n \end{bmatrix} \tag{2.19}$$

is the matrix having the first n eigenvectors. Similarly, n-dimensional representations are also obtained for the test object for comparison purposes. Usually, $n \ll N$, so that the matching complexity is considerably reduced.

Note that when only a subset of the eigenvectors is used for representations, then the reconstructed image based on that subset will not be the same as the original one. It can be shown that PCA, or KLT, will yield the best reconstruction in the mean square sense among all transformations, i.e. KLT is an optimal transform in the mean square sense.

2.3 Principal Component Analysis Using Neural Networks

If all the data is available *a priori*, then PCA can be performed by using conventional techniques like singular value decomposition (SVD). This method can also be developed by using neural networks [45], however it finds all the eigenvectors even when only the first few are needed. Where feature extraction is concerned, usually only a few of the principal components are needed. Hebbian learning rule can be used in neural networks to extract only a subset of all the principal components.

2.3.1 Hebbian learning rule

Hebbian learning has been widely used as a basic building block of associative memory in a variety of unsupervised learning applications. It is based on the explicit rule of synaptic modification first proposed by Donald Hebb [46]

> *When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.*

In other words, the synaptic strengths of neurons is adjusted in proportion to the activity of the pre- and post-synaptic neurons.

In neural learning rules, usually the training vectors are presented one after the other. Thus the input, output and weight changes are functions of time. Let the input vector at a discrete instant of time k be given by

$$\mathbf{x}(k) = \begin{bmatrix} x_1(k) & x_2(k) & \cdots & x_n(k) \end{bmatrix}^T, \tag{2.20}$$

and the weight vector at the same time be given by

$$\mathbf{w}(k) = \begin{bmatrix} w_1(k) & w_2(k) & \cdots & w_n(k) \end{bmatrix}^T, \tag{2.21}$$

and, the single-output by

$$y(k) = \mathbf{x}^T(k)\mathbf{w}(k). \tag{2.22}$$

Then the simple Hebbian algorithm may be expressed as

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \eta \mathbf{x}(k)y(k). \tag{2.23}$$

where η is the learning constant. The neural network is shown in Figure 2.2. As the

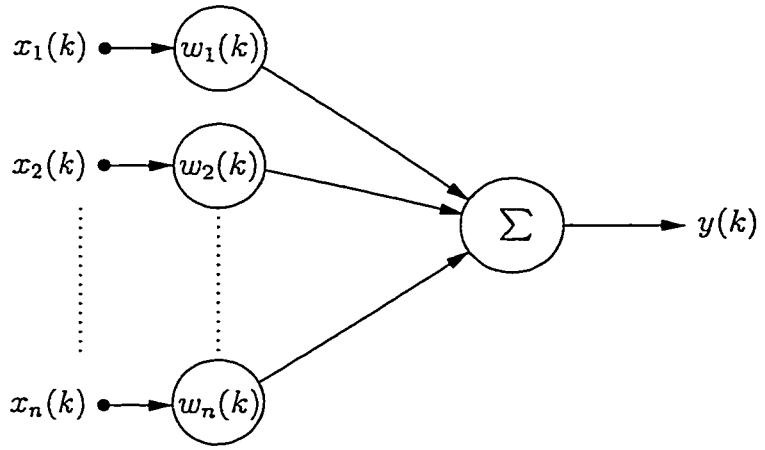


Figure 2.2: Neural network for the Hebbian learning rule. The network has a feed-forward unsupervised structure. The Hebbian algorithm extracts the first principal component at the output, y, as $k \to \infty$.

training progresses, the weight vector, $\mathbf{w}(k+1)$, approaches the direction of the first principal component. The major drawback in this learning rule is that the length of the weight vector is unbounded. This can be remedied by a re-normalizing its length after each step and bounding it within unity, giving the following algorithm [47]

$$\bar{\mathbf{w}}(k+1) = \mathbf{w}(k) + \eta \mathbf{x}(k)y(k) \tag{2.24}$$

$$\mathbf{w}(k+1) = \frac{\bar{\mathbf{w}}(k+1)}{\|\bar{\mathbf{w}}(k+1)\|}. \tag{2.25}$$

and the weight vector asymptotically approaches the eigenvector corresponding to the largest eigenvalue, i.e.

$$\mathbf{w}(k) \to \mathbf{q}_1 \qquad \text{as } k \to \infty \tag{2.26}$$

Moreover, if the learning constant, η, is sufficiently small, the above two equations can be approximated to give a single adjustment rule, also known as "Oja's Rule" [47]

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \eta y(k)\big[\mathbf{x}(k) - \mathbf{w}(k)y(k)\big]. \tag{2.27}$$

Based on the Oja's Rule other networks have been proposed which have more than one neuron. One of the popular ones is the Sanger's Generalized Hebbian Algorithm (GHA) [48]. In this algorithm, the weights for the first output are arranged according to Oja's Rule. The updates of the successively next neurons are performed by not considering the affect of the previous neurons. Thus, forcing the next neurons to extract different principal components.

The GHA algorithm is given by

$$\mathbf{w}_j(k+1) = \mathbf{w}_j(k) + \eta y_j(k)\big[\mathbf{x}(k) - \bar{\mathbf{x}}_j(k)\big]. \tag{2.28}$$

where

$$\bar{\mathbf{x}}_j(k) = \sum_{i=1}^{j} \mathbf{w}_i(k) y_i(k) \tag{2.29}$$

The neural network is shown in Figure 2.3. In GHA, the principal components are extracted in a hierarchical manner. While the outputs are calculated from the original data, the successive outputs learn subsequent principal components by subtracting estimates of earlier components from the input before the data reaches the learning algorithm. This results in the first neuron learning the most dominant principal component, and the next neuron the next dominant principal component, and so on.

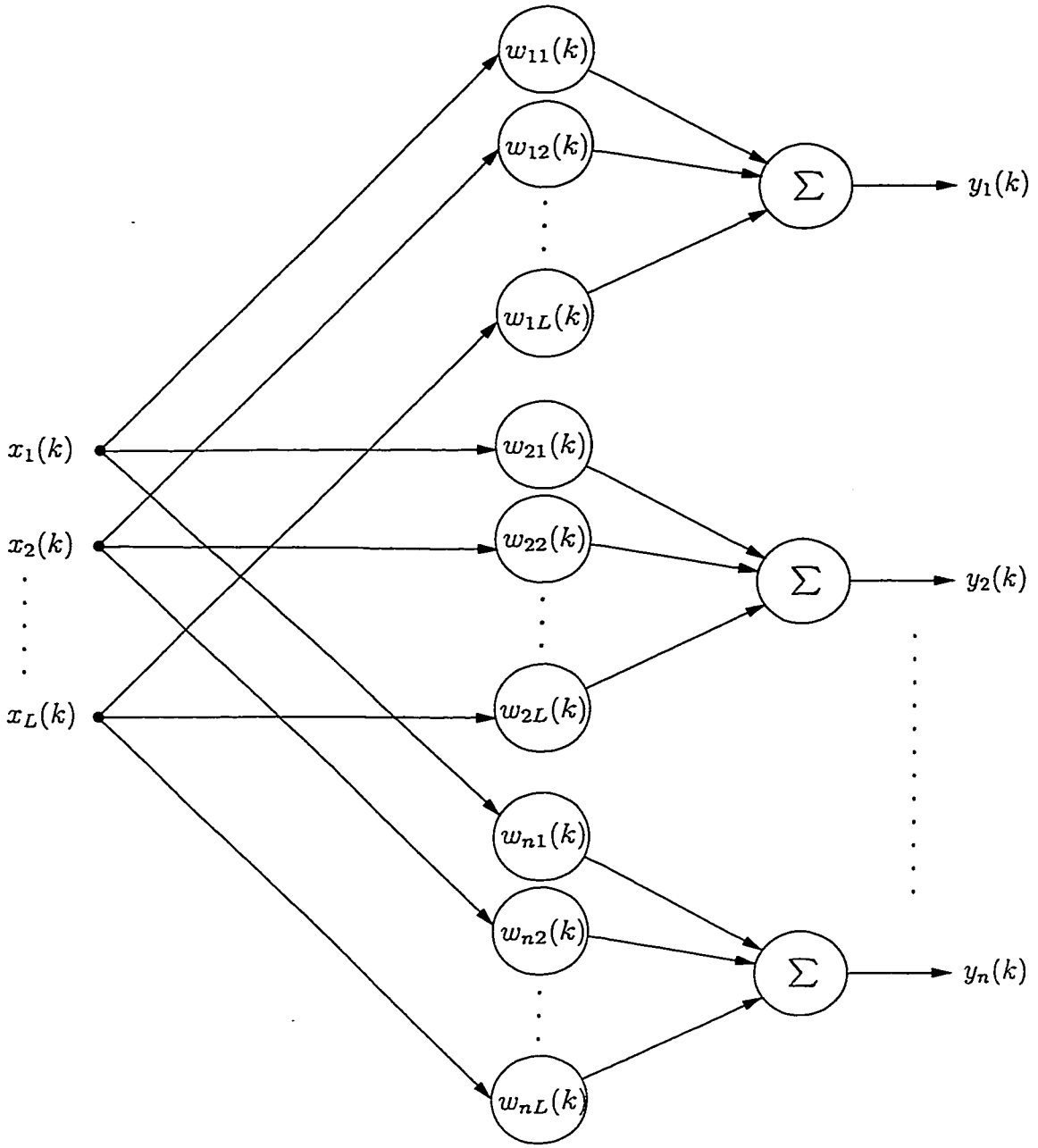


Figure 2.3: Neural network which extracts the first n principal components using GHA. Principal components are extracted in a hierarchical order, with the first neuron extracting the most dominant component, and second neuron extracting the next dominant component and so on.

2.4 Conclusion

In this chapter, the theory behind PCA was introduced. It was also shown how unsupervised neural networks can be used to extract features from training data based on second order structure. The main point in the PCA is the assumption that all features are dependent on first and second order statistics only. This method assumes that higher order statistics do not exist, and if they do exist, they are simply ignored. In natural scenes this could present problems since there is significant amount of higher order statistical information in them. Thus, it is expected that methods that take all orders of statistics into consideration are expected to extract more meaningful features. Independent component analysis is a method which tries to extract statistics of all order. The theory behind independent component analysis is given in the next chapter.

Chapter 3

Independent Component Analysis

Recent years have seen a renewed interest in unsupervised learning algorithms for finding more reliable and less complex representations of data. The most recent one being researched upon is Independent Component Analysis (ICA). ICA was originally developed a few years ago in relation with blind source separation (BSS) problems, in which the goal is to extract independent source signals from their linear mixtures using minimum of *a priori* information. BSS techniques have applications in such areas as communications, speech processing, medical signal processing, financial forecasting, image processing, and also in feature extraction. Representation of data is a main problem in neural networks and signal processing research and applications. In most of circumstances the main objective is transforming the input data using a suitable transformation such that the transformed data is represented in a manner that facilitates subsequent analysis.

The outline of this chapter is as follows. The basics of ICA are presented in Sections 3.1 and 3.2. ICA is a relatively new field and it has more than one definition. These are given in Section 3.3. In Section 3.4 the data model used to implement ICA is explained. Some of the algorithms that have been recently proposed in the literature are reviewed in Section 3.5. It is also explained why ICA is considered to represent features of the image in an efficient manner. At end of

the present chapter, in Section 3.6, the algorithm for the implementation of ICA is presented.

3.1 Background information for ICA

ICA attempts to extract independent components from the input signal. This raises the question of defining independence first. The information needed for this purpose lies in the statistical description of random variables given by moments, joint moments, p.d.fs, etc.

3.1.1 Moments of a random variable

Let X be a continuous random variable[1]. The cumulative distribution function (c.d.f) is given by

$$F_X(x) = P(X \leq x) \tag{3.1}$$

and its probability distribution function (p.d.f) is given by

$$p_X(x) = \frac{dF_X(x)}{dx} \tag{3.2}$$

The random variable is completely characterized either by its c.d.f. or by its p.d.f. But in practice neither the c.d.f nor the p.d.f. are known. To overcome this problem, *moments* and *central moments* of the random variable are used, even though the resulting description of the random variable may not be completely accurate.

[1]In the terminology commonly used in probability and stochastic processes theory, a random variable is denoted by an uppercase Latin character, and its value is denoted by a lowercase Latin character. Additionally, here a multidimensional random variable is denoted by a bold uppercase Latin character and its value is denoted by a bold lowercase Latin character.

The n-th order moment[2] of a random variable is defined as

$$\mu_X(n) = E\{X^n\} = \int_{-\infty}^{+\infty} x^n p_X(x)\, dx. \tag{3.3}$$

The first order moment is called the *mean* of a random variable, and is given by

$$\mu_X = E\{X\} = \int_{-\infty}^{+\infty} x p_X(x)\, dx. \tag{3.4}$$

The n-th order central moment of a random variable X is defined as

$$m_X(n) = E\{(X - \mu_X)^n\} = \int_{-\infty}^{+\infty} (x - \mu_X)^n p_X(x)\, dx. \tag{3.5}$$

The second order central moment, given by

$$\sigma_x^2 = m_X(2) = E\{(X - \mu_X)^2\} = \int_{-\infty}^{+\infty} (x - \mu_X)^2 p_X(x)\, dx, \tag{3.6}$$

is called the variance of the random variable and it measures the scatter of the variable. The third order central moment measures the skewness of the p.d.f. about its mean. The fourth order central moment measures the steepness or the flatness of the p.d.f. It is also called *kurtosis*, *excess* or *excess coefficient*. Random variables that have a steeper (more peaked) p.d.f. than a Gaussian random variable, have positive kurtosis, and are called leptokurtic, and the ones that have flatter p.d.f. than a Gaussian random variable have negative kurtosis, and are called platykurtic. Moments higher than two dealing with the higher order statistics are called higher order moments of a random variable.

Similarly, more than one random variable can also be considered simultaneously. The joint probability distribution of m random variables $X_1, X_2, \ldots, X_m$ is given by

$$F_{X_1, X_2, \ldots, X_m}(x_1, x_2, \ldots, x_m) = P(X_1 \leq x_1, X_2 \leq x_2, \cdots, X_m \leq x_m,) \tag{3.7}$$

[2]The characteristics of a random variable, or a distribution, are usually denoted by a Greek character.

and the joint p.d.f. is given by

$$p_{X_1, X_2, \ldots, X_m}(x_1,\ x_2,\ \ldots,\ x_m) = \frac{\partial^m F_{X_1, X_2, \ldots, X_m}(x_1,\ x_2,\ \ldots,\ x_m)}{\partial x_1 \partial x_1 \cdots \partial x_m} \tag{3.8}$$

A number of random variables are said to be independent if and only if their joint p.d.f. is the product of their individual p.d.fs., i.e.

$$p_{X_1, X_2, \ldots, X_m}(x_1,\ x_2,\ \ldots,\ x_m) = p_{X_1}(x_1) p_{X_2}(x_2) \cdots p_{X_m}(x_m) \tag{3.9}$$

Just as scalar random variables have been described above, multidimensional random vectors can also be defined in the same way. Similar terms can be defined in case of a multidimensional random vector, $\mathbf{X}$, given by

$$\mathbf{X} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_m \end{bmatrix}, \tag{3.10}$$

where the random variables $X_1,\ X_2,\ \ldots,\ X_m$ are its components. The c.d.f. of $\mathbf{X}$ is then given by

$$F_{\mathbf{X}}(\boldsymbol{x}) = P(X_1 \leq x_1 \cap X_2 \leq x_2 \cdots \cap X_m \leq x_m) \tag{3.11}$$

and its probability distribution function (p.d.f) is given by

$$p_{\mathbf{X}}(\boldsymbol{x}) = \frac{\partial^n F_{\mathbf{X}}(\boldsymbol{x})}{\partial x_1 \partial x_2 \cdots \partial x_m}. \tag{3.12}$$

The n-th order moment of the random vector is given by

$$\mu_{\mathbf{X}}(n) = \int_{-\infty}^{\infty} \mathrm{x}^n p_{\mathbf{X}}(\boldsymbol{x})\ d\mathrm{x} = \begin{bmatrix} \mu_{X_1}(n) \\ \mu_{X_2}(n) \\ \vdots \\ \mu_{X_m}(n) \end{bmatrix}. \tag{3.13}$$

The Fourier transform of the p.d.f. of a random variable is called the moment generating function (m.g.f.) and is given by

$$\psi_x(\omega) = \int_{-\infty}^{\infty} e^{j\omega x} p(x)\, dx = E\{\exp^{j\omega x}\}. \tag{3.14}$$

If the m.g.f. exists for all values of ω in an interval around $\omega = 0$ then it can be shown that the n-th order derivative of the m.g.f. gives the n-th order moment of the random variable and is denoted by

$$\psi_x^n(\omega = 0) = \left.\frac{\partial^n \psi_x(\omega)}{\partial \omega}\right|_{\omega=0} = E\{x^n\}. \tag{3.15}$$

The logarithm of m.g.f. is called *cumulant generating function* (c.g.f.),

$$\phi_x(\omega) = \ln \psi_x(\omega). \tag{3.16}$$

The c.g.f of a distribution is used to define another set of descriptive constants for the distribution, given by κ_r, such that

$$\frac{\kappa_1}{1!} + \frac{\kappa_2}{2!} + \cdots \frac{\kappa_r}{r!} + \cdots = \log \psi_x(\omega). \tag{3.17}$$

By expanding the logarithm and equating the terms of same power of κ, it can be shown that the first four coefficients, in terms of moments about the origin, are given by [49]

$$\begin{aligned} \kappa_1 &= \mu(1) = 0 \\ \kappa_2 &= \mu(2) \\ \kappa_3 &= \mu(3) \\ \kappa_4 &= \mu(4) - 3\mu(2)^2. \end{aligned} \tag{3.18}$$

It was mentioned earlier that a random variable can be characterized completely either by its p.d.f. or by its c.g.f., but as the exact knowledge of both of these is very hard, and in most of the circumstances impossible to obtain, the next best way to characterize a random variable is by computing its cumulants. Generally, first four cumulants have been considered sufficient for characterizing a random variable.

3.1.2 Entropy and mutual information

In this section, the concept of entropy is presented as it plays a central role in information transfer across a network and in mutual information in random variables. Furthermore, it will become clear subsequently that, entropy and mutual information turn out to be the fundamental quantities needed to ascertain independence of various distributions. Although entropy is introduced here for completeness of the work, the reader is referred to other sources for further details [50, 51, 52].

Since the original work in information theory by Shannon, the concept of entropy has bound itself to communication and information transfer. Entropy is a fundamental measure of information theory. The theory has also been employed to develop information-theoretic models that lead to self-organization in a principled manner. In the context of the work of this thesis, information theory assumes significant importance since it is related to the *maximum mutual information principle* [31, 53, 54].

Entropy is used to measure uncertainty in a random variable, higher the uncertainty greater the information in the random variable. The most popular example to demonstrate this concept is that of tossing of a coin. If the coin is unbiased, then the probability of either heads or tails occurring is 0.5, and it is hard to guess what the next outcome will be. On the other hand, if the coin is biased then the probability of heads or tails occurring doesn't remain the same. Consider that the coin has heads on both the sides, i.e. the coin is biased to give heads all the time. Then the probability of heads occurring in the next outcome is 1, so that the next experiment need not be performed, since the outcome is known in advance. In other words, there is no information - no surprise - contained in tossing of the coin to check the next outcome. However, if the coin is unbiased, there is maximum information contained in the outcomes of the experiment, since each experiment has to be performed to know the outcome. And if the coin is biased to give more heads than tails, then the information is neither zero nor maximum, but somewhere in

between. The variation of entropy in the coin tossing experiment just described is shown in Figure 3.1.

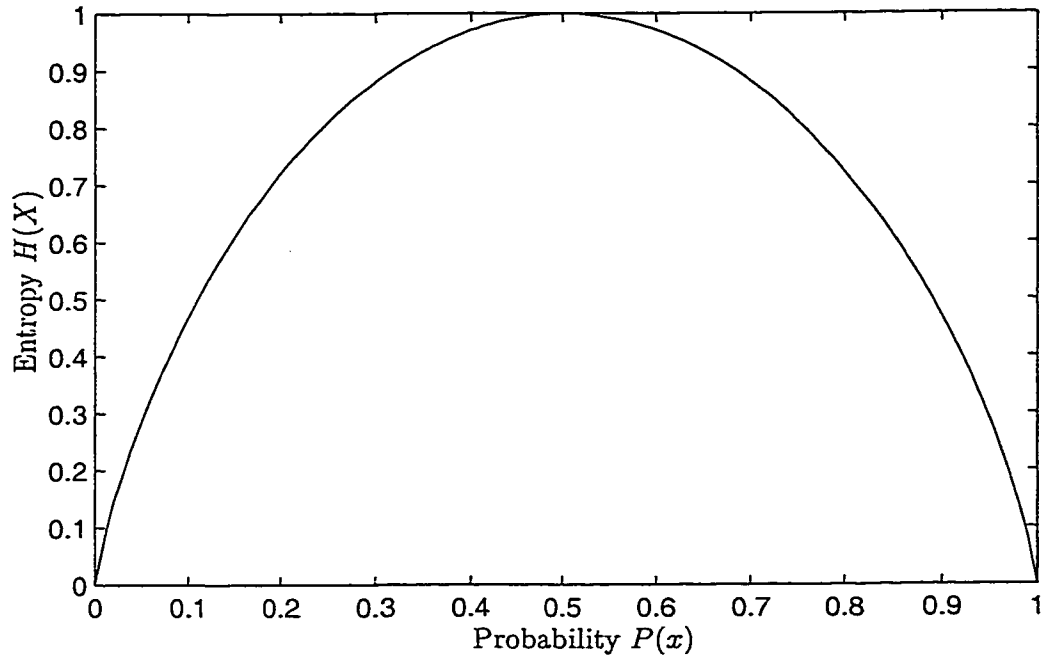


Figure 3.1: Variation of entropy in the case of a coin being tossed. The entropy is maximum when both the outcomes are equiprobable ($P(x) = 0.5$). The coins is biased when $P(x) \neq 0.5$, and the two extremes are at $P(x) = 0$ and $P(x) = 1$, when the coins is biased to give heads or tails respectively.

The entropy of a discrete random variable X with p.d.f. $p_X(x)$ is defined as

$$H(X) = -\sum_{x \in \mathcal{X}} p_X(x) \log p_X(x), \tag{3.19}$$

where $\mathcal{X}$ is the set of all possible outcomes, and $H(X)$ is called the entropy of the random variable X. The entropy is maximum when all the events of an experiment

are equally probable and it is zero when the certainty of one of the events is maximum, i.e. when one of the events has a probability of 1. Also, the lower bound of entropy is 0, i.e. $H(X) \geq 0$. Note that the log of a distribution of the random variable is used in defining the entropy. If the base of the logarithm is e, then the units of entropy are *nats*, and if the base is 2, then the units are *bits*. Also in equation (3.19), $0 \log 0$ is defined to be 0, i.e. $0 \log 0 \triangleq 0$.

If a continuous random variable is considered, then

$$H_d(X) = -\int_{-\infty}^{\infty} p_X(x) \log p_X(x)\, dx = -E\{\log p_X(x)\}, \tag{3.20}$$

is called the *differential entropy* of the continuous random variable X. Although differential entropy behaves as absolute entropy (entropy in the discrete case), it has one major difference, in the sense that differential entropy can have negative value [50].

On similar terms as above, differential entropy for a continuous random vector $\mathbf{X}$ having joint p.d.f. $p_{\mathbf{X}}(\boldsymbol{x})$ can be defined as

$$H_d(\mathbf{X}) = -\int_{-\infty}^{\infty} p_{\mathbf{X}}(\boldsymbol{x}) \log p_{\mathbf{X}}(\boldsymbol{x})\, dx = -E\{\log p_{\mathbf{X}}(\boldsymbol{x})\}. \tag{3.21}$$

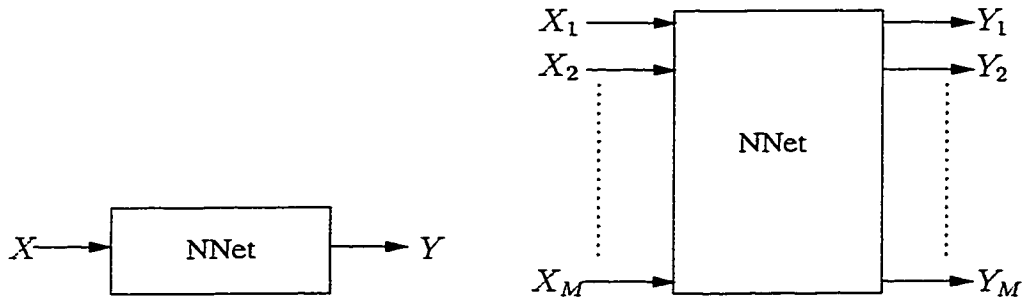
In the case of two discrete random variables, $\mathbf{X}$ and $\mathbf{Y}$, with joint p.d.f. $p_{X,Y}(x, y)$, their *joint entropy* is defined as

$$H(X, Y) = -\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p_{X,Y}(x, y) \log p_{X,Y}(x, y) \tag{3.22}$$

where $\mathcal{X}$ and $\mathcal{Y}$ are the sets of possible outcomes of random variables X and Y, respectively.

It was stated before that entropy and information play a central role in the development of self organizing neural networks based on mutual information. The network should be such that it learns input-output relationships on the basis of its input and the performance of the output. Mutual information plays an important role in this scenario, which can be imagined as a neural network having

an input and an output and the network is trying to organize itself such that it learns the uncertainty about the input after observing its output. Figure 3.2 shows the schematic of a network transforming an M-dimensional discrete random vector $\mathbf{X} = \begin{bmatrix} X_1 & X_2 & \cdots & X_M \end{bmatrix}^T$ to a random vector $\mathbf{Y} = \begin{bmatrix} Y_1 & Y_2 & \cdots & Y_M \end{bmatrix}^T$, of same dimension, at the output. In the case of a one dimensional random vector X, Figure



(a) 1-D random variable transformation.

(b) M-D random vector transformation.

Figure 3.2: Schematic of a network transforming an M-dimensional random vector, $\mathbf{X}$, to an output random vector $\mathbf{Y}$.

3.2(a), the entropy $H(X)$ is the *prior* uncertainty about the input. To measure this uncertainty by observing the output only, *conditional entropy* is defined, which is given by

$$H(X|Y) = H(X,Y) - H(Y) \tag{3.23}$$

with the property that

$$0 \leq H(X|Y) \leq H(X). \tag{3.24}$$

In the above definition, $H(X,Y)$ represents the amount of uncertainty remaining about the system input X *after* observing the system output Y. The difference

between the input entropy and the conditional entropy is the uncertainty about the input that is resolved by observing the output, and is called the *mutual information* between the input and the output and is given by

$$I(X;Y) = H(X) - H(X|Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p_{X,Y}(x,y)\log\left(\frac{p_{X,Y}(x,y)}{p_X(x)p_Y(y)}\right). \quad (3.25)$$

From the above definition, properties of mutual information can be derived. The most important ones include:

1. Mutual information between two random variables, X and Y, is symmetric.

$$I(X;Y) = I(Y;X). \quad (3.26)$$

2. Mutual information is always non-negative.

$$0 \leq I(X;Y). \quad (3.27)$$

   It is worth noting that this property is similar to the property that all distance metrics possess, it will be used to relate mutual information and a distance measure between probability distributions.

3. Entropy is a special case of mutual information, in the sense that entropy is the mutual information within the random variable itself.

$$H(X) = I(X,X). \quad (3.28)$$

4. The mutual information can also be expressed in terms of the entropy of the output

$$I(X;Y) = H(Y) - H(Y|X), \quad (3.29)$$

   where $H(Y)$ is the differential entropy of the output (conditional entropy), it is ensemble average information conveyed by the system output Y, and $H(Y|X)$ is the processing noise, or the ensemble average of information which *did not* come from the input. Note that if there is absolutely no processing noise, then $H(Y)$ is at its minimum value.

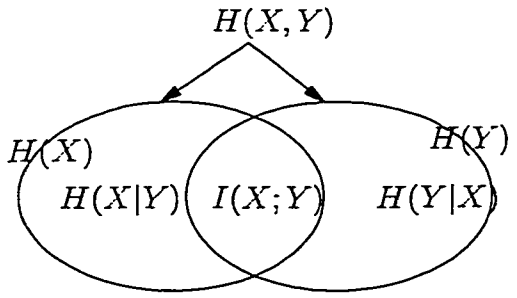


Figure 3.3: Schematic representation of the marginal entropy, conditional entropy and the mutual information.

The definition of mutual information in the case of a multi-dimensional random variable (the transformation is shown in Figure 3.2(b)), can be easily obtained by extending the definition of the one dimensional case, which results in

$$I(\mathbf{X};\mathbf{Y}) = H(\mathbf{X}) - H(\mathbf{X}|\mathbf{Y}) = \sum_{\boldsymbol{x}\in\underline{\mathcal{X}}}\sum_{\boldsymbol{y}\in\underline{\mathcal{Y}}} p_{\mathbf{X},\mathbf{Y}}(\boldsymbol{x},\boldsymbol{y}) \log\left(\frac{p_{\mathbf{X},\mathbf{Y}}(\boldsymbol{x},\boldsymbol{y})}{p_{\mathbf{X}}(\boldsymbol{x})p_{\mathbf{Y}}(\boldsymbol{y})}\right). \quad (3.30)$$

3.1.3 Independence, KL divergence, and entropy

The independence of a multidimensional random vector was defined by equation (3.9). It can be written in a compact vector form as

$$p_{\mathbf{X}}(\boldsymbol{x}) = \prod_{i=1}^{m} p_{X_i}(x_i). \quad (3.31)$$

Thus the degree of independence of a number of random variables can be verified by observing how well the joint probability distribution of the set of random variables and their individual probability distributions satisfy equation (3.31). One of the obvious ways to see this is to define some sort of a distance measure between the two sides of equation (3.31), such that the independence of the random variables is inversely propotional to the distance measure. One such distance measure is the *Kullback-Liebler (KL) divergence* [55, 56].

Let $f_{\mathbf{X}}(\boldsymbol{x})$ and $g_{\mathbf{X}}(\boldsymbol{x})$ be two different probability density functions of an m-by-1 random variable $\mathbf{X}$, then the Kullback-Liebler divergence (it is also known as cross entropy, discrimination, Kullback-Liebler entropy, etc.) between $f_{\mathbf{X}}(\boldsymbol{x})$ and $g_{\mathbf{X}}(\boldsymbol{x})$ is defined as

$$\mathcal{KL}_{f_{\mathbf{X}}||g_{\mathbf{X}}} = \int_{-\infty}^{\infty} f_{\mathbf{X}}(\boldsymbol{x}) \log\left(\frac{f_{\mathbf{X}}(\boldsymbol{x})}{g_{\mathbf{X}}(\boldsymbol{x})}\right) d\boldsymbol{x}. \tag{3.32}$$

Main properties of the $\mathcal{KL}$ divergence are:

1. $\mathcal{KL}_{f_{\mathbf{X}}||g_{\mathbf{X}}} \geq 0$.

2. It is invariant with respect to:

   (a) order of arrangements of the components in $\boldsymbol{x}$,

   (b) their amplitude scaling, and

   (c) monotonic non-linear transformation.

Property (1) above is the most interesting one in relation to the independence of random variables and thus to ICA. Notice that the equality in equation (3.32) holds if and only if $f_{\mathbf{X}}(\boldsymbol{x}) = g_{\mathbf{X}}(\boldsymbol{x})$. From the above properties, it is obvious that the $\mathcal{KL}$ divergence exhibits some distance metric characteristics, but it should not be construed as a true distance measure, since $\mathcal{KL}_{f_{\mathbf{X}}||g_{\mathbf{X}}} \neq \mathcal{KL}_{g_{\mathbf{X}}||f_{\mathbf{X}}}$.

From the definition of the KL divergence in equation (3.32) and the expression of mutual information of two random vectors in equation (3.30), the relationship between KL divergence and mutual information can be easily observed:

$$I(\mathbf{X};\mathbf{Y}) = \mathcal{KL}_{f_{\mathbf{X},\mathbf{Y}}||f_{\mathbf{X}}f_{\mathbf{Y}}}. \tag{3.33}$$

This means that the KL distance is the mutual information between the joint probability distribution and the product of their marginal distributions. This is where the importance of the relationship between the mutual information and the KL divergence becomes apparent. By using equations (3.32) and (3.30), the KL distance

between the joint probability distribution of a random vector $\mathbf{X}$, given by $p_{\mathbf{X}}$, and the product of its marginal distributions, given by $p_{\mathbf{X}}^{\prod} = \prod_{i=1}^{m} p_{X_i}(x_i)$, is given by

$$\mathcal{KL}_{f_{\mathbf{X}}||p_{\mathbf{X}}^{\prod}} = \int_{-\infty}^{\infty} f_{\mathbf{X}}(\boldsymbol{x}) \log \left(\frac{p_{\mathbf{X}}(\boldsymbol{x})}{\prod_{i=1}^{m} p_{X_i}(x_i)} \right) d\boldsymbol{x}. \tag{3.34}$$

By comparing the above expression with the condition of independence of random variables, given in equation (3.9), it is clear now that the KL divergence will be zero in equation (3.34) if and only if the random variables X_i, the components of the random vector $\mathbf{X}$, are all independent. And in the event that the random variables are not independent, then the mutual information will be positive. Thus the form of KL in equation (3.34) is an attractive tool to see the degree of independence between the components of a random vector; smaller the KL distance, greater will be the independence. This expression has been used in the application of maximization of mutual information to achieve redundancy reduction.

3.2 Criterion for Independence

In the previous section, the characteristics of random variables were introduced and the notion of independence of random variables was presented. For ICA, the reliable verification of independence is the most crucial step. Theoretically, the independence of random variables can be verified by using equation (3.9), or by computing the KL distance. The fact that the explicit knowledge of the joint p.d.fs. of a random vector as well as the marginal p.d.fs. of the individual components of the random vector is not usually known in practice severely restricts the direct applications of the equations.

An introduction of some of the main ICA algorithms is given in Section 3.5. The introductory algorithms proposed in the literature for ICA used only the first few cumulants to characterize the random process under consideration. Some of the algorithms just minimized the sum of fourth order cumulants to verify independence.

Still others used some heuristic approaches such that independence was achieved after a number of iterations.

3.3 Definitions of ICA

In the literature three definitions of linear ICA have appeared [57, 58]. These are described below. It should be noted that the definition of ICA have to take into account the fact that the underlying independent components are unknown and that the observed signal is a linear combination of those components. Also to be more general, noise must also be considered while defining ICA. But since noise is never known a *a priori*, inclusion of noise complicates ICA even further.

Definition I *ICA of a random vector* $\mathbf{x}$ *consists of finding a linear transform* $\mathbf{s} = \mathbf{W}\mathbf{x}$ *so that the components* s_i *are as independent as possible, in the sense of maximizing some function* $F(s_1, s_2, \ldots, s_n)$ *that measures independence.*

This is a general definition and no assumptions on the data are made. This is a simple definition based on the premise that the observed signal $\mathbf{x}$ is a linear combination of the independent components, $\mathbf{s}$, and that $\mathbf{W}$ is the transformation that extracts those components. This definition corresponds to the analysis formula (see Section 1.3). Also to use the above definition, a measure of independence of s_i needs to be specified.

Definition II *ICA of a random vector* $\mathbf{x}$ *consists of estimating the following generative model for the data:*

$$\mathbf{x} = \mathbf{A}\mathbf{s} + \mathbf{n} \tag{3.35}$$

where the components of $\mathbf{s}$ *are assumed to be independent,* $\mathbf{A}$ *is the mixing matrix, and* $\mathbf{n}$ *is the noise vector.*

Although this definition takes into account the noise, $\mathbf{n}$, in the signal, the noise vector complicates the problem of separating the independent components, since no *a priori* knowledge of the input data is assumed and the noise is not known. In most of the applications in current research, noise is considered just as another independent source to be estimated and is omitted from the equation, giving the following noise free definition of ICA. Furthermore, note that this formula corresponds to the synthesis formula (see Section 1.3), i.e. it constructs a signal from the independent components, instead of breaking it down as was the case in Definition I.

Definition III *ICA of a random vector* $\mathbf{x}$ *consists of estimating the following generative model for the data:*

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{3.36}$$

The above definition is used in this thesis. The details of this definition are given below in Section 3.4. In this definition, noise is not considered. This simplifies the problem of ICA since noise is considered as just another source. As was mentioned before, this is motivated from the fact that since the independent components are unknown, it cannot be determined, *a priori*, what is noise and what is useful information.

3.4 ICA Data Model

The data model used here is similar to the one used in [57]. Consider m scalar random variables $x_1, x_2, \ldots, x_m$ which are linear combinations of n *unknown independent components* of the source signal $s_1, s_2, \ldots, s_n$, assumed mutually statistically independent and zero mean. s_i may be considered as the underlying 'cause' of x_i. Let the variables x_i be represented by

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_m \end{bmatrix}^T, \tag{3.37}$$

and the variables s_i be represented by

$$\mathbf{s} = \begin{bmatrix} s_1 & s_2 & \cdots & s_n \end{bmatrix}^T; \tag{3.38}$$

then the linear combination is given by the relationship

$$\mathbf{x} = \sum_{i=1}^{n} \mathbf{a}_i s_i = \mathbf{A}\mathbf{s} \tag{3.39}$$

Here $\mathbf{A}$ is an unknown fixed $m \times n$ matrix of full rank, called the mixing matrix, whose columns are denoted by $\mathbf{a}_i$, $i = 1, \ldots, n$. These columns are called the *ICA basis vectors*, or *basis functions*. The weighting of these functions by $\mathbf{s}$ gives the image $\mathbf{x}$. These are the counterparts of the principal eigenvectors in PCA. The fundamental problem of ICA is then to recover the underlying causes $\mathbf{s}$ from the mixtures $\mathbf{x}$. Since only $\mathbf{x}$ is observed, the problem changes to estimating $\mathbf{B}$ in

$$\mathbf{s} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{x} = \mathbf{B}\mathbf{x} \tag{3.40}$$

It may be also worthwhile to mention that if the signal $\mathbf{x}$ is a function of time, then the coefficients in $\mathbf{s}$ are also a function of time, and the equation (3.39) becomes

$$\mathbf{x}(t) = \sum_{i=1}^{n} \mathbf{a}_i s_i(t) = \mathbf{A}\mathbf{s}(t). \tag{3.41}$$

As is the case in most of the transformations, a new basis is chosen for the data. The choice of the new basis determines the properties that will be possessed by the transformed data. Consider the Karhunen-Loéve transform, it is the optimal transforms in the mean square sense. In other words, data reconstructed using this transform will have the least mean squared error when compared to the original data, although the transform may not be optimal for recognition or classification. On the other hand, in the case of ICA, the basis given by the columns of $\mathbf{A}$ is such that the coefficients in $\mathbf{s}$ are as independent as possible. This means that, generally, the basis is nonorthogonal, and that higher order statistics are needed to determine

the ICA expansion. However, this kind of transformation often characterizes the fundamental properties of the data better than standard KLT. Note that unlike the case with KLT, ICA is not optimal in the mean square sense, but it could be better for recognition purposes.

So from the definition given in equation (3.39), the object of performing ICA on an observed data is to find a transformation in which the components s_i are statistically as independent from each other as possible. Since no *a priori* information is available either of the source signals or of the mixing matrix, it may seem impossible to extract the independent components. But it turns out that by making certain assumptions the problem can indeed be solved.

3.4.1 Assumptions in the data model

In the above data model, it is assumed the the source signals are zero mean and mutually statistically independent. In practice, it has been found that the natural source signals are indeed usually independent. Consider the example of the cocktail party problem in which the sources are the various people speaking in the room at the same time. It is highly improbable that the voices of various people are dependent upon one another. This problem has been dealt with at many places in the literature and the voices have been separated successfully, thus supporting the assumption that the voices are independent. It is also assumed that the original sources are unobservable and all that is available are the n linear mixtures $\mathbf{x}$.

It is further assumed that the number of available different mixtures m is at least as large as the number of sources n, i.e. $n \leq m$. Also, each source signal s_i is a stationary zero-mean stochastic process. Since it is impossible to separate several Gaussian sources from each other [57], only at most one of the source signals s_i is allowed to have a Gaussian distribution. Also, the mixing matrix must be of full rank. Last of all, it is assumed that the observed signals do not have noise in them, because it is usually impossible to separate the noise from the data. Since the noise

is not known in advance, it is usually considered just as another independent source and is separated along with the rest of the other sources. In other words, noise is separated out as one of the sources in s.

Having mentioned some of the assumptions used in practice to perform ICA, attention is now turned to preprocessing the data.

3.4.2 Preprocessing of the data

Prior to applying the data to the algorithm to perform ICA, it is sphered. Although sphering may not be necessary for all kinds of data or in various algorithms, it has been observed that sphering speeds up the convergence rate even when it is not strictly needed as a preprocessing stage. The process of 'sphering' the data involves removing the mean from the data and passing it through a zero phase whitening filter. In other words, the row means are removed from the input data matrix. Let the data be represented by $\mathbf{X}_{raw}$. First we get $\mathbf{X}_0$ as the zero-mean data and subsequently filtering it with a zero-phase whitening filter, $\mathbf{W}_z$, where

$$\mathbf{X}_o = \mathbf{X}_{raw} - E\{\mathbf{X}_{raw}\}, \text{ and } \mathbf{W}_z = 2\left(\sqrt{E\{\mathbf{X}_o\mathbf{X}_o^T\}}\right)^{-1} \tag{3.42}$$

we get the sphered data, $\mathbf{X}$, given by

$$\mathbf{X} = \mathbf{W}_z\mathbf{X}_o. \tag{3.43}$$

Sphering essentially removes the first and second order characteristics which helps in speeding up of the convergence of the ICA algorithm [59]. Sphering makes the covariance matrix of the input data equal to $4\mathbf{I}$ and makes the transformation roughly orthogonal, i.e. $\mathbf{W}^T\mathbf{W} = \mathbf{I}$. After sphering, the data is ready to be given to the ICA algorithm.

3.5 Approaches to ICA

It was mentioned in Section 3.4 that the ICA problem was to estimate $\mathbf{B}$ in equation (3.40). There have been many approaches proposed to achieve the ICA objective. The main aspect in the various methods is to achieve independence, and they all differ in the particular method used to do this. In a very broad sense, the approaches can be categorized into two groups as

1. the ones exploiting cumulants, and
2. the ones which use information theoretic principles and do not use cumulants.

3.5.1 Comon's ICA algorithm

Pierre Comon is noted for his work which dealt with theoretical aspects of ICA in a systematic manner [57]. His approach was based on using cumulant up to the fourth order to achieve independence. Moreover, his objective was to find an independent set of basis vectors, and not on source separation. But this can be related to ICA by considering the new bases as the set of sources which are weighted and combined to produce the observed data.

Due to computational reasons, Comon used cumulants up to the fourth order cumulant. But even by considering the first four cumulants, and by neglecting higher order statistics, the algorithm is still computationally very intensive and involved. Nevertheless, his work opened doors to further research in ICA.

The algorithm proposed by Comon was an iterative one, and not based on neural networks. Many neural networks would be introduced in the literature in the next few years which would perform better and still be computationally less complicated.

3.5.2 Jutten-Herault algorithm

Jutten and Herault were among the first ones to use neural methods to achieve ICA [58], and they called theirs a *neuromimitic* approach.

Their approach used an unmixing equation of the form

$$\mathbf{u}(t) = \mathbf{x}(t) - \mathbf{W}(t)\mathbf{u}(t) \tag{3.44}$$

where $\mathbf{u}$ is the output vector, $\mathbf{x}$ is the input to the network, $\mathbf{W}$ is the weight matrix, and t is the time variable. In this method, the weight matrix is restricted to be square. The algorithm consists of cancelling the nonlinear cross-correlations of the output by updating the weight matrix according to the following relationship

$$\Delta W_{ij} \propto f(u_i) \cdot g(u_j), \quad i \neq j \tag{3.45}$$

to give the output

$$\mathbf{u} = (\mathbf{I} + \mathbf{W})^{-1}\mathbf{x} \tag{3.46}$$

where $f(\cdot)$ and $g(\cdot)$ are two different, odd, nonlinear functions fixed throughout the training process. Under these conditions, the diagonal terms are set to zero and the output after convergence produces the independent components. This algorithm has one serious drawback, the convergence is subject to some severe restrictions [60]. Other work related to use of nonlinear cross-correlation cancellation for ICA can be found in [61, 62, 63, 64, 65, 66]. The algorithm given in [67, 63] is particularly important, since it does not involve the inversion of the weight matrix, and the weight update expression is given by

$$\Delta \mathbf{W} \propto \Big(\mathbf{I} - g_1(\mathbf{y})g_2(\mathbf{y}^T)\Big)\mathbf{W} \tag{3.47}$$

where the functions $g_1(\cdot)$ and $g_2(\cdot)$ are applied separately to every component of $\mathbf{y}$.

3.5.3 Neuron models for ICA by Hyvärinen and Oja

Hyvärinen and Oja gave various kinds of neural models for ICA using kurtosis [68]. Recall that a sub-Gaussian signal has negative kurtosis and a super-Gaussian signal has positive kurtosis. Hyvärinen and Oja gave single neuron models to extract an independent component of negative kurtosis by minimizing a contrast function, and to extract an independent component of positive kurtosis by maximizing the contrast function. They further developed the algorithm to give a general contrast function which can be minimized or maximized depending on the sign of kurtosis of the independent components. Moreover, they have also modified algorithms for sphered data as well as for non-sphered data.

3.5.4 Information maximization approach

In their seminal paper, Bell and Sejnowski [69] used the information as a cost function. Their approach was different from the previous ones, and also attractive, because it completely bypassed the need to compute cumulants and moments. They used mutual information instead of cross-statistics to achieve ICA solutions. Their approach is also known as the info-max approach.

Bell's original algorithm [69] had a serious drawback. It involved the weight matrix inversion. This problem was solved by using *natural gradient* [70, 71, 72], resulting in an elegant info-max algorithm that has been used successfully in many applications including feature extraction, bio-medical signal processing, and face recognition. This approach is used in this thesis for view based 3-D object recognition. The details of this algorithm are given in the next section.

3.6 Info-Max algorithm

The info-max algorithm utilizes the maximization of information criterion for ICA. It is implemented on a feed-forward neural network shown in Figure 3.4. The M-dimensional signal $\mathbf{x}$ is the input to the network. Recall that the input is considered to be made up of independent components s_i (see Section 3.3). Each input x_i is weighted by the weights connected to the neuron and summed to give the M outputs u_i, i.e.

$$u_i = \sum_{j=1}^{M} w_{ij} x_j \; ; \quad i = 1, 2, \cdots, M \tag{3.48}$$

which may be expressed in following compact form

$$\mathbf{u} = \mathbf{W}\mathbf{x}. \tag{3.49}$$

Each u_i is then passed through a monotonic non-decreasing nonlinearity to obtain the outputs y_i

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{bmatrix} = \begin{bmatrix} f(u_1) \\ f(u_2) \\ \vdots \\ f(u_M) \end{bmatrix}, \tag{3.50}$$

which in the vector form may be written as

$$\mathrm{y} = f(\mathrm{u}) = f(\mathbf{W}\mathrm{x}). \tag{3.51}$$

The joint of the output of the network is now given by

$$H(y_1, y_2, \cdots, y_M) = H(y_1) + H(y_1) + \cdots H(y_M) - I(y_1, y_2, \cdots, y_M), \tag{3.52}$$

where $H(y_i)$ are the marginal entropies and $I(y_1, y_2, \cdots, y_M)$ is the mutual information in the output. It is obvious from the above expression that maximization

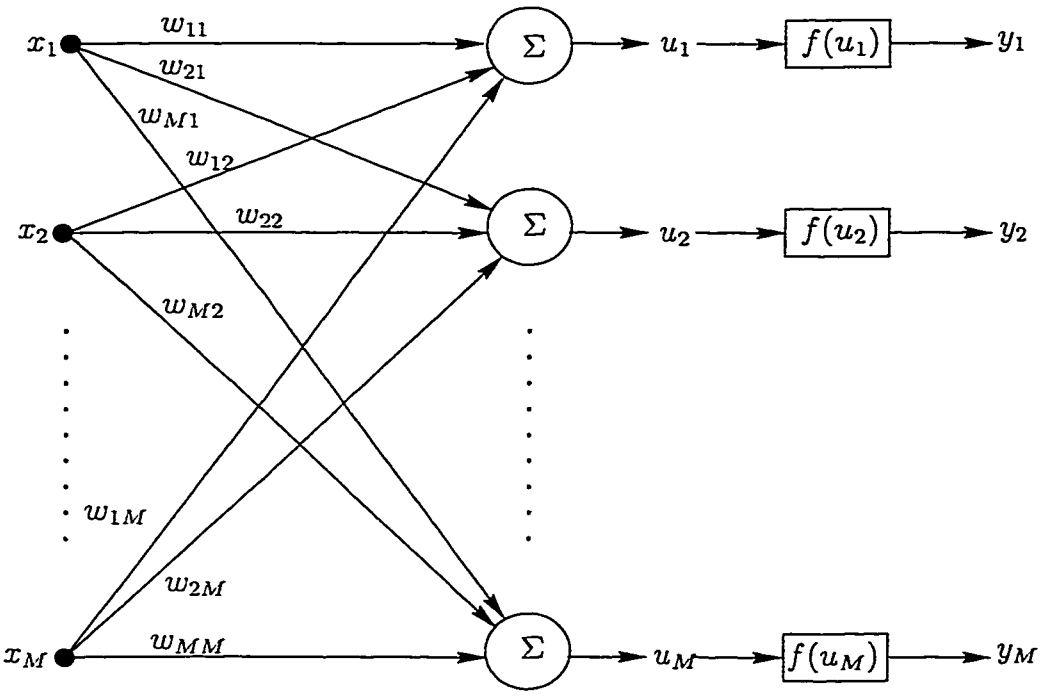


Figure 3.4: The neural network used for implementing the Independent Component Analysis. The weights are updated using the info-max algorithm. For simplicity, the feedback loop from the outputs to the weights is not shown.

of joint entropy can be achieved by maximizing the marginal entropies and by minimizing the mutual information. In the ideal case, the mutual information would be zero and the marginal distribution of the outputs is uniform, since the marginal entropy of an amplitude bounded random variable is maximum when its marginal distribution is uniform. In other words, the nonlinearity $f(\cdot)$ used in the network should have the shape of the c.d.f. of the source s_i. In the info-max algorithm, the nonlinearity is fixed, and the marginal entropy is maximized with respect to the weight matrix $\mathbf{W}$. Before proceeding with the details of the algorithm, a few words need to be said about the choice of the nonlinearity $f(\cdot)$.

It should be noted that in the algorithm no attempt is made to change the

nonlinear function. The adaptation and learning involves only the change of the weight matrix according to a specified cost function. Thus, one should not get a false impression that the focus is concentrated only on the weights of the network and the nonlinearity is not significant here. The significance of the nonlinearity comes into play right in the beginning when it is fixed. For the marginal entropys to be maximum, the marginal distributions of all the outputs should be uniform, this implies that the derivative of the nonlinear logistic function should match the true source distribution of s_i [59, 73]. But then this would mean that for the best possible results, the source distributions should be known in advance (in this case the problem of ICA would not be too difficult, since the independent sources' characteristics would already be known), which in practice is not the case. The next best thing is to choose a suitable logistic function which satisfies the above criterion. It has been reported that the logistic sigmoid function behaves appropriately when applied to natural signals such as sounds and images. It is given by

$$\mathrm{y} = \frac{1}{1 + \exp^{-\mathrm{u}}}. \tag{3.53}$$

Recalling the definition of the marginal entropy from Section 3.1.2, the sum of marginal entropies in equation (3.52) can be written as

$$H(\mathrm{y}) = -\sum_{i=1}^{M} E\{\log p(y_i)\} - I(\mathrm{y}) \tag{3.54}$$

where $I(\mathrm{y}) \triangleq I(y_1, y_2, \ldots, y_n)$.

Since the nonlinear mapping given by $f(\cdot)$ is a monotonic increasing mapping, the density function $p(y_i)$ can be written in terms of the input density $p(u_i)$ as [74] (Note that the monotonic increasing (or decreasing) requirement is a necessary condition in order to have a one to one relationship)

$$p(y_i) = \frac{p(u_i)}{|\frac{\partial y_i}{\partial u_i}|}. \tag{3.55}$$

Thus, equation (3.54) becomes

$$H(\mathbf{y}) = -\sum_{i=1}^{M} E\left\{\log \frac{p(u_i)}{|\frac{\partial y_i}{\partial u_i}|}\right\} - I(\mathbf{y}) \tag{3.56}$$

The objective is to adapt the weight matrix in such a way so as to maximize the output joint entropy. The update rule for this purpose can be obtained by taking the gradient of equation (3.56) with respect to the weight matrix $\mathbf{W}$, resulting in the following expression

$$\frac{\partial H(\mathbf{y})}{\partial \mathbf{W}} = -\frac{\partial}{\partial \mathbf{W}}\left(\sum_{i=1}^{n} E\left\{\log \frac{p(u_i)}{|\frac{\partial y_i}{\partial u_i}|}\right\}\right) + \frac{\partial}{\partial \mathbf{W}}(-I(\mathbf{y})) \tag{3.57}$$

It can be again seen how the shape of the nonlinearity affects the maximization of the joint entropy. If the c.d.f. of the source density is equal to the shape of the nonlinearity, it will result in the source density being equal to the derivative of the nonlinearity, thus making the first part of the right hand side of equation (3.57) equal to zero. In this ideal case the mutual information is minimized and the joint entropy is maximized. As was previously mentioned, sigmoidal function has been found to behave satisfactorily in most of the circumstances involving natural data. The equation (3.57) also supports the relationship between maximization of the joint entropy and minimization of mutual information, each of these logically leads to the other. Also recall that the minimization of mutual information results in the independence of the sources, hence maximization of joint entropy also results in independence of the sources. Also recall that the extraction of independent components is related to the principal of redundancy reduction. With these remarks, the basis for info-max algorithm has been established. What remains now is to determine how to change the weight matrix so as to achieve maximum joint entropy of the outputs.

The joint entropy of the outputs can be written in the vector form as (see Section 3.1.2)

$$H(\mathbf{y}) = -E\{\log p(\mathbf{y})\} \tag{3.58}$$

Also, since the input $\mathbf{x}$ is related to the output $\mathbf{y}$ by a monotonic non-decreasing transformation according to equation (3.51), the density function, $p(\mathbf{y})$ of the output is thus related to the density function of the input, $p(\mathbf{x})$, as

$$p(\mathbf{y}) = \frac{p(\mathbf{x})}{|\frac{\partial \mathbf{y}}{\partial \mathbf{x}}|} = \frac{p(\mathbf{x})}{|J|} \tag{3.59}$$

where $|J|$ is the absolute value of the determinant of the Jacobian matrix $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$. Thus equation (3.58) can be written as

$$\begin{aligned} H(\mathbf{y}) &= -E\{\log p(\mathbf{y})\} && (3.60)\\ &= -E\Big\{\log \frac{p(\mathbf{x})}{|J|}\Big\} && (3.61)\\ &= -E\{\log p(\mathbf{x})\} + E\{\log |J|\} && (3.62)\\ &= H(\mathbf{x}) + E\{\log |J|\} && (3.63) \end{aligned}$$

The gradient of equation (3.63) with respect to the weight matrix results in

$$\frac{\partial H(\mathbf{y})}{\partial \mathbf{W}} = \frac{\partial}{\partial \mathbf{W}}(E\{\log |J|\}), \tag{3.64}$$

where $\frac{\partial H(\mathbf{x})}{\partial \mathbf{W}} = 0$, since $H(\mathbf{x})$ is constant and does not depend on the weights. It may also be considered as the *a priori* uncertainty in the input which has no relation to the neural weights.

The determinant of the Jacobian can be evaluated as

$$\begin{aligned} |J| &= \Big|\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\Big| \\ &= \Big|\frac{\partial \mathbf{y}}{\partial \mathbf{u}}\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\Big| && \text{(chain rule)} \\ &= \Big|\frac{\partial \mathbf{y}}{\partial \mathbf{u}}\Big|\Big|\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\Big| && (3.65) \end{aligned}$$

where $\frac{\partial \mathbf{y}}{\partial \mathbf{u}}$ and $\frac{\partial \mathbf{u}}{\partial \mathbf{x}}$ are Jacobian matrices. The last step follows from the fact that determinant of a product of matrices is the product of their determinants [75].

Now,

$$\left|\frac{\partial \mathbf{y}}{\partial \mathbf{u}}\right| = \begin{vmatrix} \frac{\partial y_1}{\partial u_1} & \frac{\partial y_1}{\partial u_2} & \cdots & \frac{\partial y_1}{\partial u_M} \\ \frac{\partial y_2}{\partial u_1} & \frac{\partial y_2}{\partial u_2} & \cdots & \frac{\partial y_2}{\partial u_M} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial y_M}{\partial u_1} & \frac{\partial y_M}{\partial u_2} & \cdots & \frac{\partial y_M}{\partial u_M} \end{vmatrix} = \begin{vmatrix} \frac{\partial y_1}{\partial u_1} & 0 & \cdots & 0 \\ 0 & \frac{\partial y_2}{\partial u_2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \frac{\partial y_M}{\partial u_M} \end{vmatrix}$$

$$= \prod_{i=1}^{M} \frac{\partial y_i}{\partial u_i} \tag{3.66}$$

where the partial derivatives are obtained by noting that there are no couplings among the neurons in Figure 3.4. Furthermore,

$$\left|\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\right| = \begin{vmatrix} \frac{\partial u_1}{\partial x_1} & \frac{\partial u_1}{\partial x_2} & \cdots & \frac{\partial u_1}{\partial x_M} \\ \frac{\partial u_2}{\partial x_1} & \frac{\partial u_2}{\partial x_2} & \cdots & \frac{\partial u_2}{\partial x_M} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial u_M}{\partial x_1} & \frac{\partial u_M}{\partial x_2} & \cdots & \frac{\partial u_M}{\partial x_M} \end{vmatrix} = \begin{vmatrix} w_{11} & w_{12} & \cdots & w_{1M} \\ w_{21} & w_{22} & \cdots & w_{2M} \\ \vdots & \vdots & \vdots & \vdots \\ w_{M1} & w_{M2} & \cdots & w_{MM} \end{vmatrix}$$

$$= \det(\mathbf{W}) \tag{3.67}$$

Using equations (3.64) to (3.67), the change of output entropy with respect to the change in the weights takes the following form

$$\frac{\partial H(\mathbf{y})}{\partial \mathbf{W}} = \frac{\partial}{\partial \mathbf{W}} E\{\log \det(\mathbf{W})\} + \frac{\partial}{\partial \mathbf{W}} \sum_{i=1}^{M} \log\left|\frac{\partial y_i}{\partial u_i}\right| \tag{3.68}$$

$$= \frac{(\mathrm{adj}\mathbf{W})^T}{\det \mathbf{W}} + \frac{1}{\frac{\partial y_i}{\partial u_i}} \frac{\partial^2 y_i}{\partial^2 u_i} x_j \tag{3.69}$$

$$= (\mathbf{W}^T)^{-1} + \frac{\frac{\partial p(\mathbf{u})}{\partial \mathbf{u}}}{p(\mathbf{u})} \mathbf{x}^T \tag{3.70}$$

where it is assumed that the nonlinearity has the shape of the c.d.f. of the source, i.e. $p(u_i) = \frac{\partial y_i}{\partial u_i}$.

Consequently, the change in weights is given by

$$\Delta \mathbf{W} \propto (\mathbf{W}^T)^{-1} + \left(\frac{\frac{\partial p(\mathbf{u})}{\partial \mathbf{u}}}{p(\mathbf{u})}\right) \mathbf{x}^T. \tag{3.71}$$

This was the original rule given by Bell and Sejnowski. But matrix inversion being a costly operation, the weight inversion was a major drawback for sources more than two or three. This can be avoided by using the natural gradient algorithm [70, 71], which essentially involves the post multiplication of equation (3.71) to give

$$\Delta \mathbf{W} \propto \left(\mathbf{I} + \left(\frac{\frac{\partial p(\mathbf{u})}{\partial \mathbf{u}}}{p(\mathbf{u})} \right) \mathbf{u}^T \right) \mathbf{W} \tag{3.72}$$

The block diagram depicting the algorithm is shown in Figure 3.5. Thus the update rule adapting the weight matrix takes the following form

$$\mathbf{W}(k+1) = \mathbf{W}(k) + \eta \Big(\mathbf{I} - \varphi(\mathbf{u}(k)) \mathbf{u}^T(k) \Big) \mathbf{W}(k) \tag{3.73}$$

where $-\frac{\frac{\partial p(\mathbf{u})}{\partial \mathbf{u}}}{p(\mathbf{u})} \triangleq \varphi(\mathbf{u})$, η is the learning constant, and k is the time index.

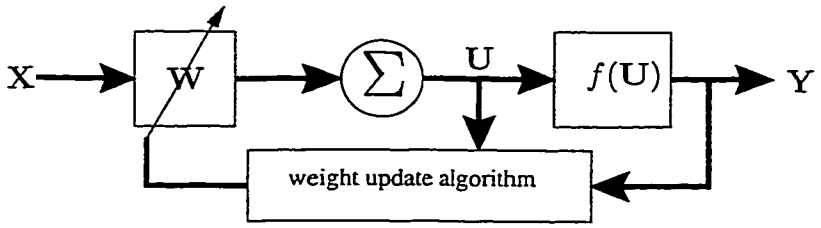


Figure 3.5: The block diagram of the neural network with the feedback loop used for updating of the weight matrix $\mathbf{W}$. The input vector has images as its rows.

The signal flow graph for this algorithm is shown in Figure 3.6 where it is shown that the input signal is being filtered by the changed weight matrix w.

3.7 Conclusion

In this chapter the basic theory behind ICA was introduced. The problem statement was explained and the definitions of ICA were also introduced. The various

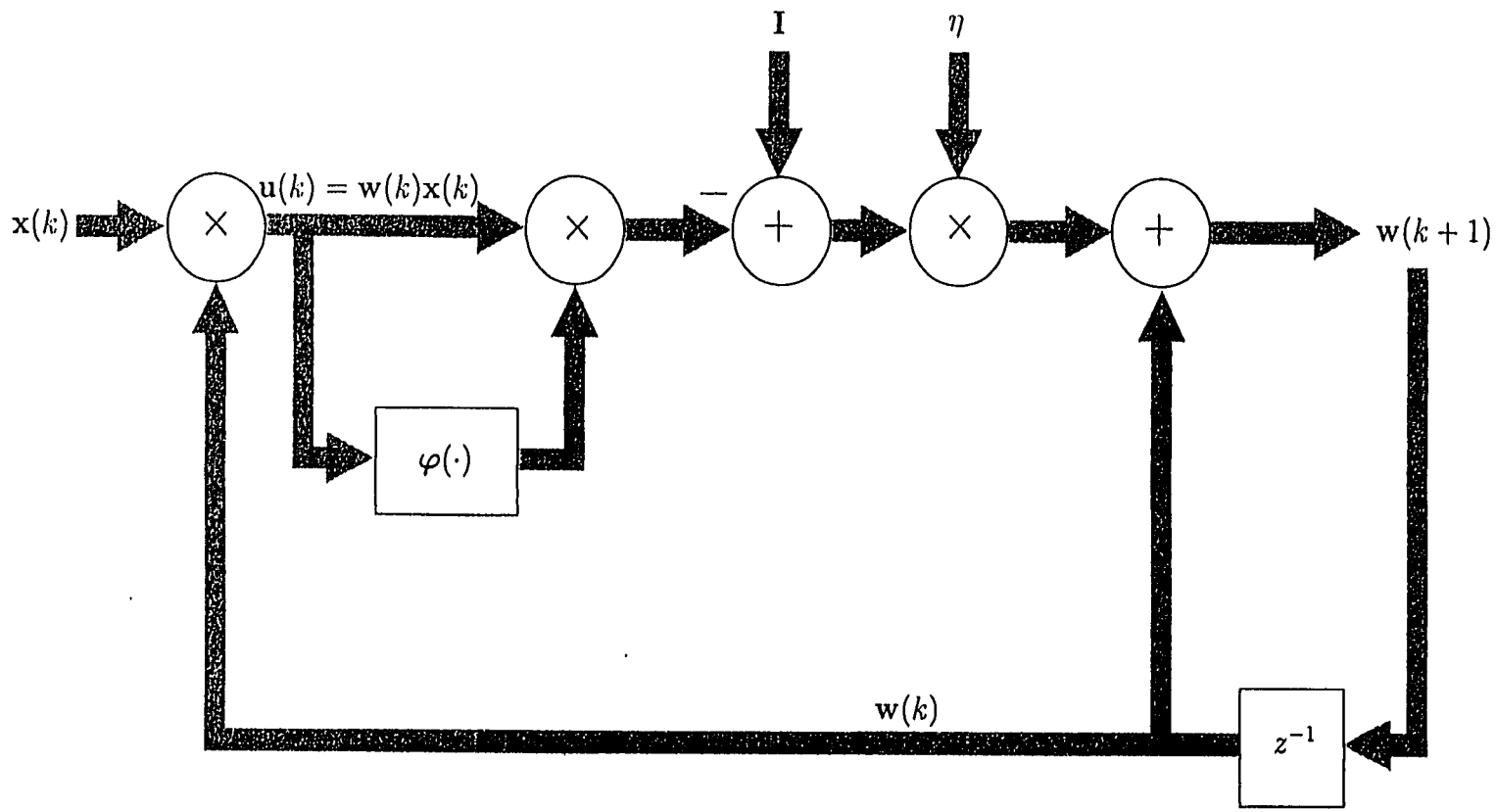


Figure 3.6: Signal flow graph of neural network implementing the info-max algorithm. The delay, z^{-1}, is for all the weights. Here $\mathbf{I}$ is the identity matrix, η is the learning constant, and k is the discrete time instant.

assumptions were also mentioned that are needed for the application of ICA. To implement ICA, Info-Max algorithm was also explained. The interesting thing about Info-Max algorithm is its simplicity and the fact that it tries to consider statistics of all orders unlike other algorithms which consider statistics only till the fourth order. In the next chapter, results of the application of ICA to 3D appearance based object recognition are presented.

Chapter 4

Experimentation and Results

4.1 Introduction

In this chapter, the results of the application of ICA and its comparison with the application of PCA for 3D object recognition are presented.

The theory behind the ICA info-max algorithm was presented in the last chapter. The theory for PCA was presented in Chapter 2. Before giving the results and comparisons of the application of both methods, the experimental setup is explained first. In Section 4.2, the first set of data used in the simulations is explained. Section 4.3 explains how the eigenspace associated with the database is constructed. In Section 4.4, the presentation of data to the ICA network is explained, along with the need to compress that data by using PCA. The results of the application of ICA to the first image database are given in Section 4.5. Then, in Section 4.6, the second database on which ICA was applied is explained and the results are presented.

4.2 The Database

The database used was the same as the one that has been used for appearance based object recognition in [20, 26]. Although full details about the database can be found

elsewhere [76], the main points are summarized below.

The images of the objects were taken using a 25mm Sony CCD camera. Ambient light was used to avoid strong shadows. Strong shadows tend to change the appearance of the object since the area under the shadow appears to be black and the intensity in that area does not give the right information about the object. In all, 20 objects were used. Each object's images were taken by placing it on a turntable. The images were taken at every 5° of pose angle, i.e. the object was rotated and an image was taken after every 5° of rotation from 0° to 360° degrees. This gave 72 images of each object, and 1440 total number of images. Thus the pose angle was the manifold parameter and the set of images gave the complete image manifold.

The images taken by the camera were cropped and rescaled to 128×128, while retaining the aspect ratio, and were histogram stretched such that the maximum pixel value was 255. The images had uniform black background and there was no occlusion.

The size of the images was changed to 64×64 due to memory constraints. The 0° pose angle view of the 20 images are shown in Figure 4.1. Note that 64×64 is a big enough size to demonstrate the effectiveness of the object recognition algorithm.

4.3 Constructing the Eigenspace of the Database

To construct the eigenspace of the database, a few of the images were chosen as the training images. Recall that the representations of images make a manifold with the pose angle as the parameter in the high dimensional eigenspace. Thus the image manifold was sampled at regular intervals of pose angle to make the training database.

If the pose angle parameter is a continuous variable, the manifold will be a smooth curve. Since it is not possible to capture images with the pose angle as a

Figure 4.1: The 20 objects database used in the 3-D appearance based object recognition.

continuous variable, the manifold will appear as a piecewise continuous curve, and if the pose angle is fine enough, the curve will appear to be smooth. The manifold for one of the objects in the database is shown in Figure 4.2.

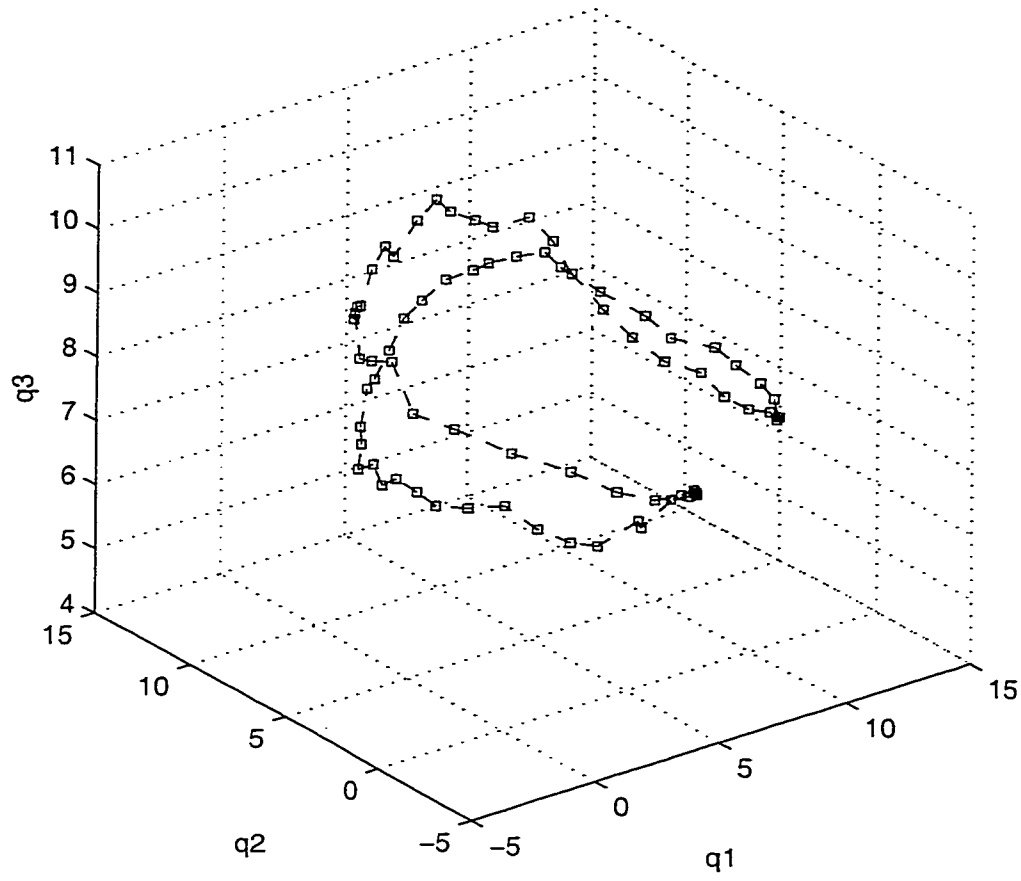


Figure 4.2: The appearance manifold of one of the objects. The eigenspace was constructed by using all the images available, i.e. images sampled as every 5°, from 0° to 360°. The curve appears smooth with the markers showing the location of the eigenimage in the 3-dimensional eigenspace with q_1, q_2, and q_3 being the three most dominant eigenvectors respectively. Also, the curve seems closed, since the last view of the object is almost the same as the first one (their poses are different by only 5°).

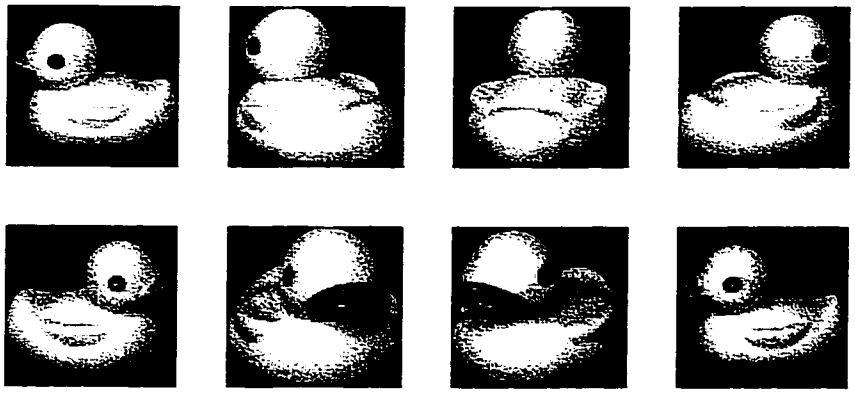


Figure 4.3: The training views of one of the objects. The pose angle was sampled at every 50°.

In the first experiment, images separated by 50° in pose angle were chosen to construct the representative eigenspace of the database. This means, every 10-th image from the database was chosen as the training image. The training views of one of the objects are shown in Figure 4.3. The representation of three different objects in the training set is shown in Figure 4.4. Another set of experiments was performed on a database consisting of objects' images sampled at every 25°. To again show how the appearance of the training objects changed from one training view to another, all the training poses of the object is shown in Figure 4.5. The representation of the training views is shown in Figure 4.6.

The images that were not in the training set were considered as test images, thus making a total of 1280 test images.

While performing PCA on the images , the images were arranged as columns of the image matrix X (see equation (2.2), Section 2.2). In this case, each image was considered as an observation and each pixel value in each image (or each row element in a column of X) was considered as a measurement. PCA makes a new set of axes on which the data can be projected such that the axes are along the

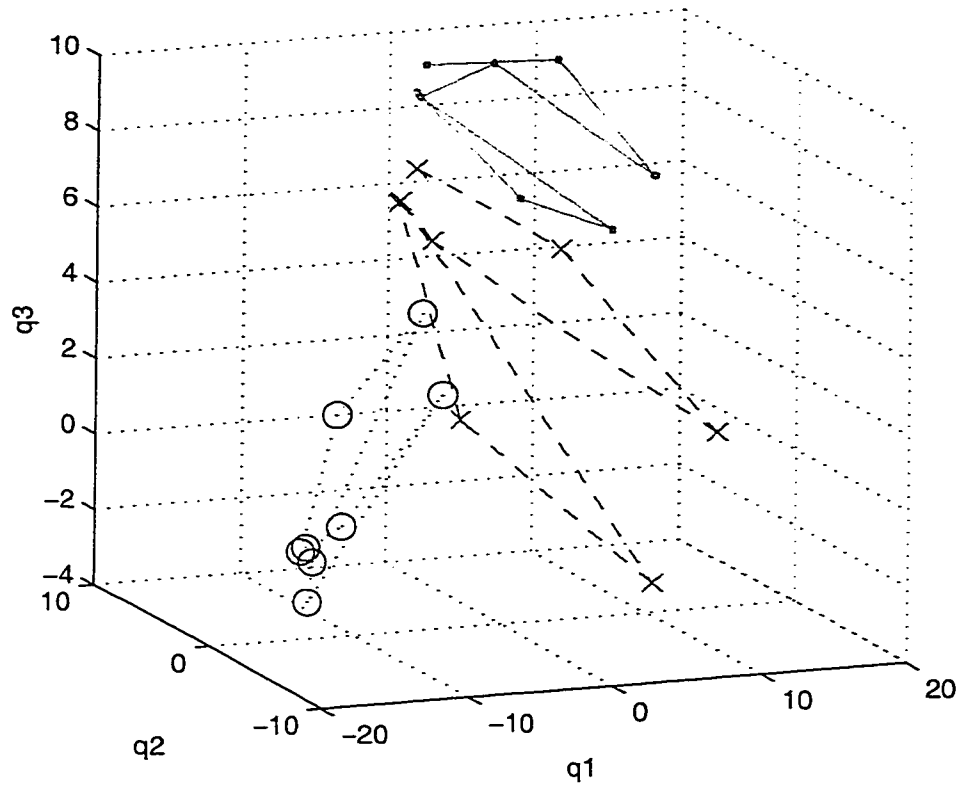


Figure 4.4: The eigenspace manifolds for three three different objects in the database with the pose angle manifold being sampled at 50°. The crosses, circles, and the dots represent the positions of the training views of the three different objects in the three dimensional eigenspace with q_1, q_2, and q_3 as the three axis.

maximum variance of the corresponding pixels in the images. PCA removes the pairwise correlations between the pixels of the images.

4.4 ICA Data Presentation to the Neural Network

In the info-max algorithm, the weight matrix is square. Hence, the network tries to extract as many components as the number of mixtures being given to it. Recall

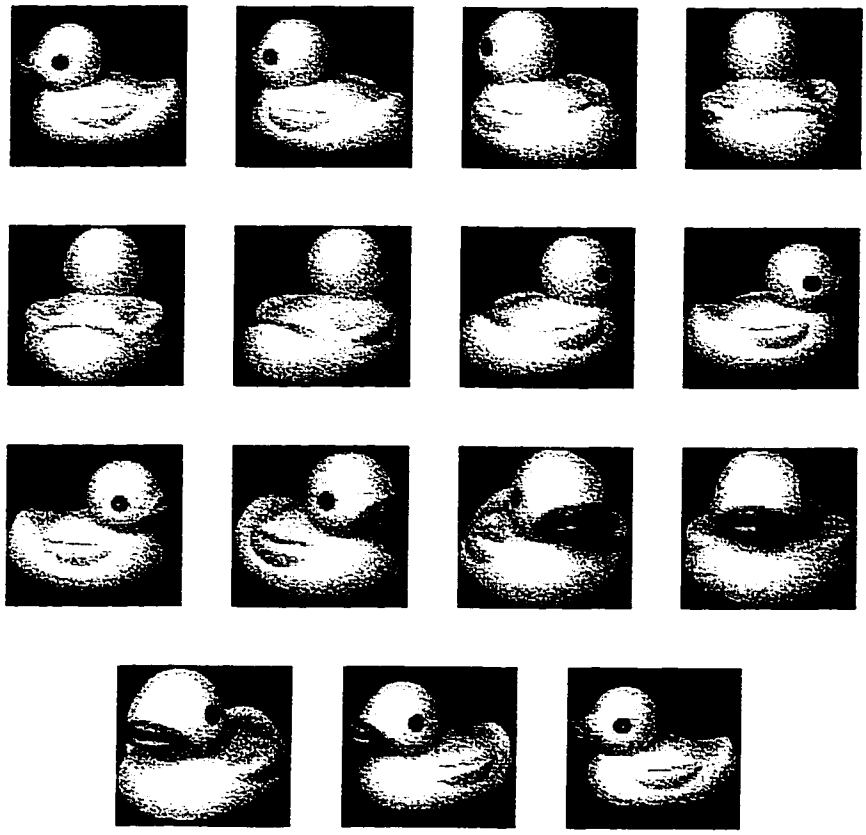


Figure 4.5: The training views of one of the objects. The pose angle was sampled at every 25°. In other words, every fifth object in the database was used as the training image.

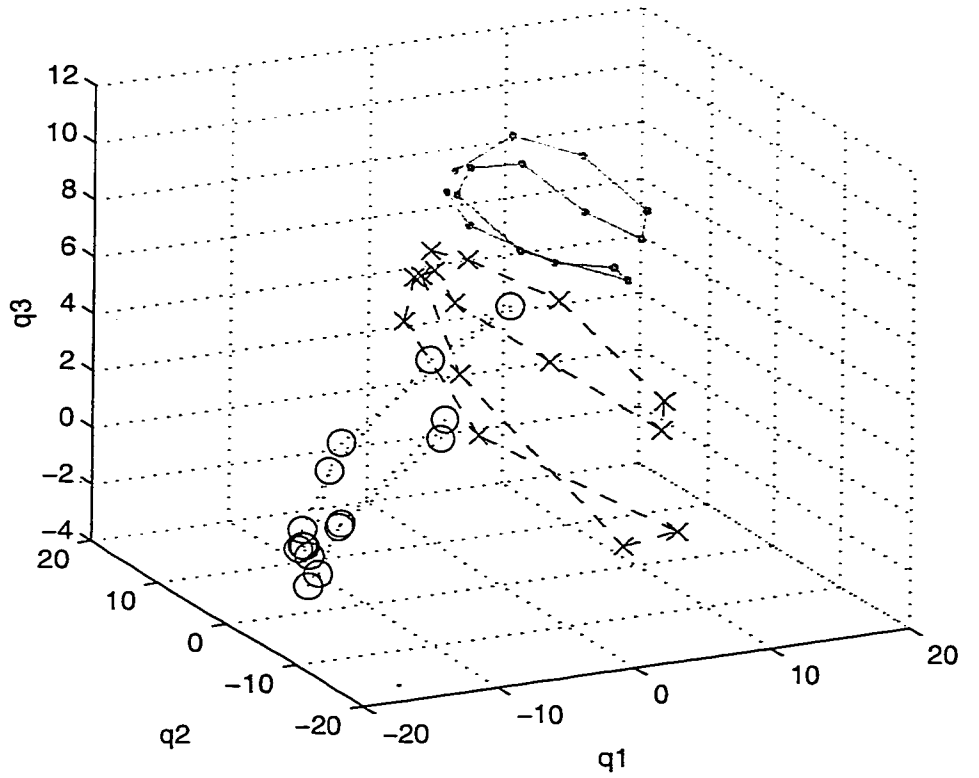


Figure 4.6: The eigenspace representation of training views of three objects. The pose angle was sampled at every 25°, or every fifth object in the database. The crosses, circles, and the dots represent the positions of the training views of three different objects in the three dimensional eigenspace with q_1, q_2, and q_3 as the three axis.

that in Chapter 1 it was mentioned that the desired goal of a visual system is that it learns independent features that make up the visible scene. The number of mixtures that may be considered is not dependent on the ICA algorithm, but the ICA algorithm is heavily dependent on the number of mixtures. Moreover it is not necessary that the number of independent components in the data be the same as the number of mixtures. For example, consider that there are 100 mixtures being presented to the network. The info-max algorithm will try to extract as many independent components. But in reality, there may be lesser number of components, say for example 50. This raises the question that how can one decide *a priori* how

many components to look for? Unfortunately, there is currently no satisfactory answer to this question. Various methods can be adopted to estimate the number of independent components in a given data, including heuristics and trail and error. In this thesis, PCA was used to make such an estimate. This was motivated by the fact that the energy content corresponding to the eigenvectors obtained by using PCA gives an approximate indication of the number of components that one has to look for. For 64×64 images, most of eigenvectors corresponding to the lower eigenvalues can be considered as noise, since most of energy is concentrated only in the first few eigenvectors. In our experiments eigenvectors numbering from the first 5 upto the first 100 were used to demonstrate the performance of the algorithm.

First, PCA was performed on the training database. The first n eigenvectors were selected and given as the rows of the input matrix to the ICA network. This is justified by the fact that the eigenvectors are also another linear mixture of the input images. Note that the input data is assumed to be some linear mixture of the underlying independent components, and therefore it can be replaced by any other linear mixture.

Another important and related advantage of using PCA was that it reduced the dimensionality of the input data. The first n dominant principal component axes (eigenvectors), out of a total of N, where N is the total number of pixels in each of the images, were given as rows of $\mathbf{X}^T$, such that the input to the ICA neural network was $\mathbf{Q}_n^T$. It should also be noted that higher order statistical characteristics still existed in the eigenvectors, since PCA removed only the second order characteristics. Also, before giving the data to the ICA network, it was sphered with the whitening filter $\mathbf{W}_z$ (see Section 3.4.2 for details).

With the input to the ICA as $\mathbf{Q}_n{}^T$, i.e. the first n eigenvectors as the rows of the input matrix, the ICA transformation expression can be written as

$$\mathbf{W}_L \mathbf{W}_Z \mathbf{Q}_n{}^T = \mathbf{U} \tag{4.1}$$

where $\mathbf{W}_L$ is the square weight matrix learned by the network and $\mathbf{U}$ has independent images in its row. The learned weight matrix and the whitening filter can be combined into one filter by the following equation

$$\mathbf{W} = \mathbf{W}_L \mathbf{W}_Z \tag{4.2}$$

where $\mathbf{W}$ yields the combined filter. Thus, equation (4.1) becomes

$$\mathbf{W}{\mathbf{Q}_n}^T = \mathbf{U} \tag{4.3}$$

or

$${\mathbf{Q}_n}^T = \mathbf{W}^{-1}\mathbf{U} \tag{4.4}$$

Also recall that data reconstruction from PCA was given as (see Section 2.2)

$$\mathbf{X}^T = {\mathbf{\Gamma}_n}^T \mathbf{Q}^T \tag{4.5}$$

where $\mathbf{\Gamma}_n$ was the representation of the database based on the first n dominant eigenvectors.

Using equations (4.4) and (4.5), the reconstructed data from the ICA output $\mathbf{U}$ could be written as

$$\mathbf{X}^T = {\mathbf{\Gamma}_n}^T \mathbf{W}^{-1}\mathbf{U} \tag{4.6}$$

Thus, the coefficients for the original images are obtained as

$$\rho = {\mathbf{\Gamma}_n}^T \mathbf{W}^{-1} \tag{4.7}$$

This yielded the coefficients for the training data in ρ. Coefficients for the test data, ρ_t, were obtained in a similar fashion by using the PCA representations of the test data, $\mathbf{\Gamma}_{nt}$, using the same number of principal components as were used in the training data, such that

$$\rho_t = {\mathbf{\Gamma}_{nt}}^T \mathbf{W}^{-1} \tag{4.8}$$

4.5 Results and Observations

The data was presented to the ICA network as described in the previous section. The recognition was done by finding the minimum distance between the ICA coefficients of a test image and the coefficients of the database images. The image in the database that was nearest to the test image coefficient vector was chosen as the recognized image.

The eigenvectors were given to the network as the rows of the input matrix. This resulted in the output matrix, $\mathbf{U}$, having images in its rows. The network attempted to extract characteristics from these images such that the corresponding pixels in them were as independent as possible. The images were of size 64×64, making an image vector 4096 elements long. The PCA performed on these images resulted in eigenvectors of the same length. The output of the ICA network was also a set of images of the same length. Thus $\mathbf{U}$ was also 4096 elements long. It should be noted that although the *rows* of $\mathbf{U}$ were made as independent as possible by the network, there was no constraint on the coefficient matrix. The coefficients had no relationship among themselves. The coefficients were the weights which determined the significance of a particular underlying independent image in the observed image. This can be seen in the following manner.

Consider the input to the ICA to be $\mathbf{X}^T$, the weight matrix as $\mathbf{W}$, and the output to be $\mathbf{U}$. Then the ICA transformation can be written as

$$\mathbf{X}^T = \mathbf{W}^{-1}\mathbf{U} = \mathbf{K}\mathbf{U} \tag{4.9}$$

where the square matrix $\mathbf{K}$, with elements k_{ij} contains the coefficients as elements. Further, consider the size of $\mathbf{X}^T$ as $N \times L$, i.e. N images of length L. Consequently, it is expected that ICA extracts $\mathbf{K}$ weights of size $N \times N$, making $\mathbf{U}$ an $N \times L$ matrix of the same size as input. The role of the coefficients can be clearly seen if the above equation is written in an expanded form as

$$\begin{bmatrix} \mathbf{X}_1{}^T \\ \mathbf{X}_2{}^T \\ \vdots \\ \mathbf{X}_L{}^T \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{N} k_{1i}\mathbf{u}_i{}^T \\ \sum_{i=1}^{N} k_{2i}\mathbf{u}_i{}^T \\ \vdots \\ \sum_{i=1}^{N} k_{Ni}\mathbf{u}_i{}^T \end{bmatrix}, \tag{4.10}$$

where $\mathbf{X}_i$ is the i-th image, and $\mathbf{u}_i{}^T$ is the i-th row of $\mathbf{U}$. Thus, the coefficients in the rows of $\mathbf{K}$ determined the magnitude of the contribution of a particular independent component to the overall observed image.

4.5.1 Info-max algorithm parameters used in the experiments

The info-max algorithm was used to the train the network under various parameter settings. Since there is no established method yet to estimate or guess the suitable choice of the parameters, their choice was based more or less on trial and error. In unsupervised neural networks, such approach is routinely used. Even in conventional adaptive filtering methods, many times the parameters in the algorithm are chosen by trial and error.

The learning rate constant η : After various experiments, it was observed that the learning constant if not kept sufficiently small could cause the algorithm to become unstable, in other words, the weights' changes become progressively large and unbounded. Proper convergence of the algorithm is marked by the weight changes decreasing progressively, until a stable point is reached.

In the case of the database in which the pose angles was sampled at 50°, initial learning constant of 0.01 caused the weights to become unstable. It was observed that $\eta = 0.001$ caused the algorithm to progress satisfactorily.

Number of iterations and epochs: The network behaved as expected if the data was presented around 1500 times to the network. Each time the data was

shuffled randomly to prevent cyclic learning of the weight matrix. It was also observed that even when the weights seemed to have stabilized around the first 400 iterations, more data presentations were necessary for further learning. Also, during each presentation, called an epoch, a block of data was given to the network. This helped speed up the execution of the program.

Weight changes and annealing: It was observed that the change in weights was maximum in the first few iterations. After that the changes showed erratic behavior as if they were oscillating around a particular point in the entropy space. To stabilize the weight vector the learning rate was decreased after a certain number of iterations. This is somewhat similar to what is done by simulated annealing in neural networks. This helps the weights to stabilize around a value as the training continues.

Number of objects in the database: The number of objects to be used in the database proved to problematic. The recognition was erratic if only a few objects were used. For example, with 5 objects in the database, no significantly consistent difference in recognition performance was observed. But by using all the 20 objects, differences in the performance were observed.

4.5.2 Results with pose angle sampling at every 50 degrees

The training database consisted of images of all objects. The pose angle manifold was sampled at 50°. The views of one of the object are shown in Figure 4.3.

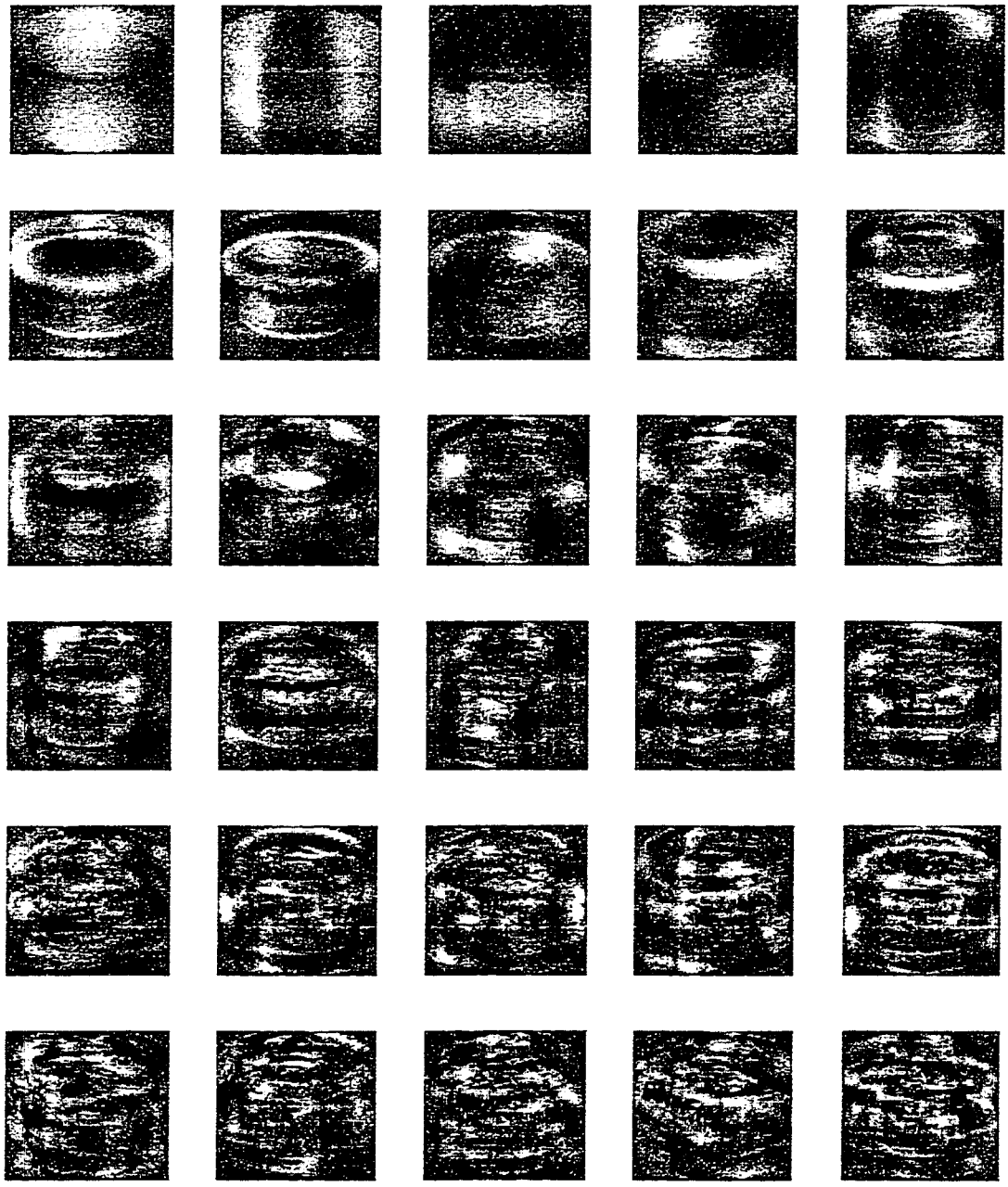


Figure 4.7: The 30 dominant eigenvectors of the training database with the pose angle being sampled every 50°. The eigenvectors are ordered from left to right, top to bottom, with the top left eigenimage corresponding to the most dominant eigenvector, and the bottom right image that of the least dominant eigenvector.

The training was done by giving various number of most dominant eigenvectors to the network. The first thirty dominant eigenvectors are shown in Figure 4.7. For comparison with eigenimages, 30 independent components are shown in Figure 4.8. The performance of the network is shown in Figure 4.9. The learning rate was changed after a set number of iterations to simulate annealing in the neural network. This is shown in Table 4.1. Best performance was observed when 25 to 35 eigenvectors were used for training. The success rate of both ICA and PCA started to fall off beyond the use of 50 eigenvectors. Although the difference in the performance is small, but it can be seen that ICA did give better recognition than PCA. The results are shown in Figure 4.9. The number of correct recognitions are shown in Table 4.2

Number of Epochs	Learning constant η
800	0.001
400	0.0005
200	0.00005
100	0.00001

Table 4.1: The learning constant used for the predetermined set of epochs. The total number of epochs were 1500. The learning rate was gradually decreased to simulate annealing in the network learning.

It is interesting to note that if more than 45 eigenvectors were used, ICA started to perform worse than PCA. This could be attributed to the actual number of underlying independent components present in the database being considered. Perhaps there were only 35 significant independent components, and the components beyond this number were not significant, or maybe they were just noise. In view of the assumption that the image is made up of independent components, it could be

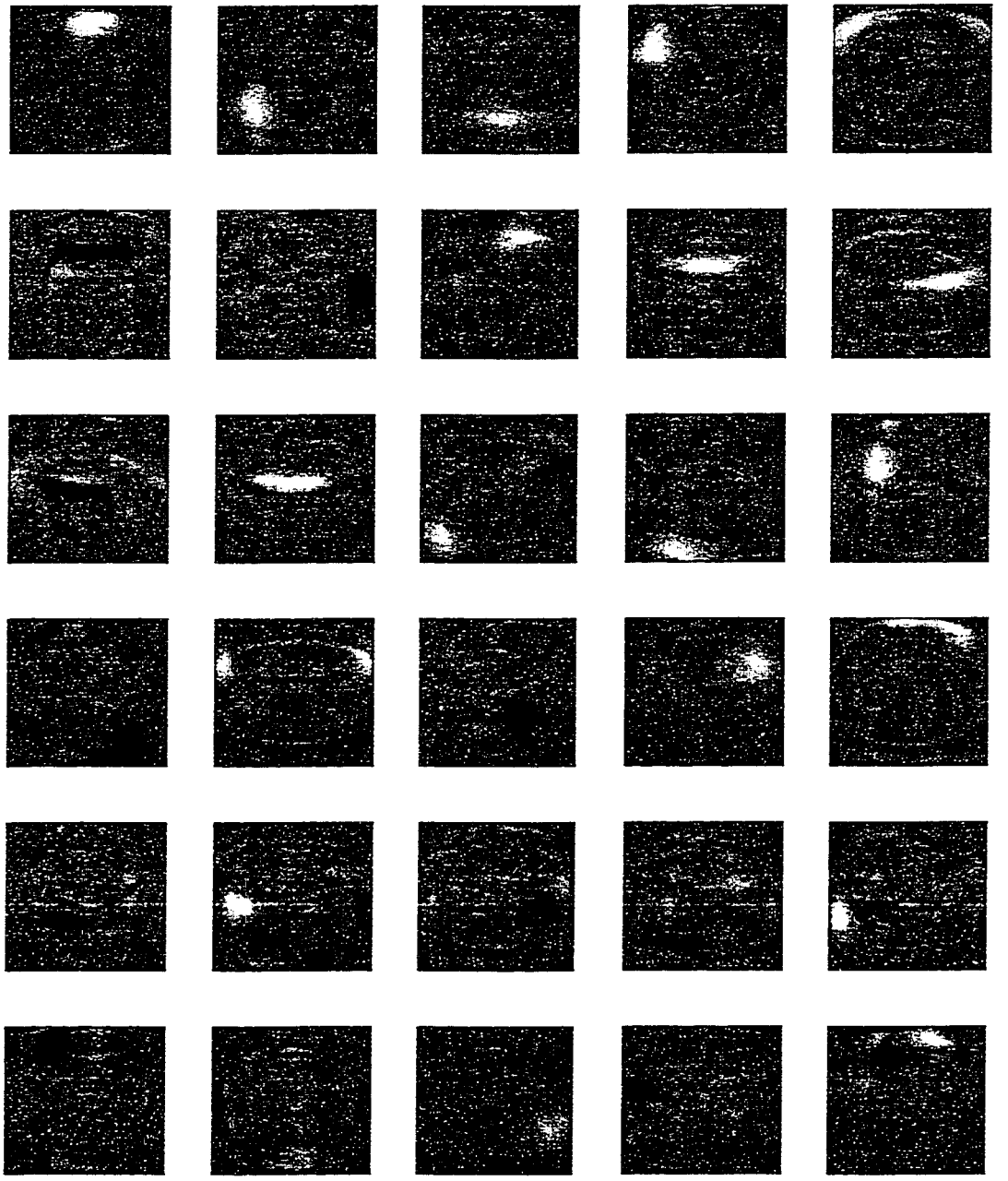


Figure 4.8: The 30 independent components as extracted by the network in the case where the pose angle was sampled every 50° to make the training database. The network does not extract the components in any order.

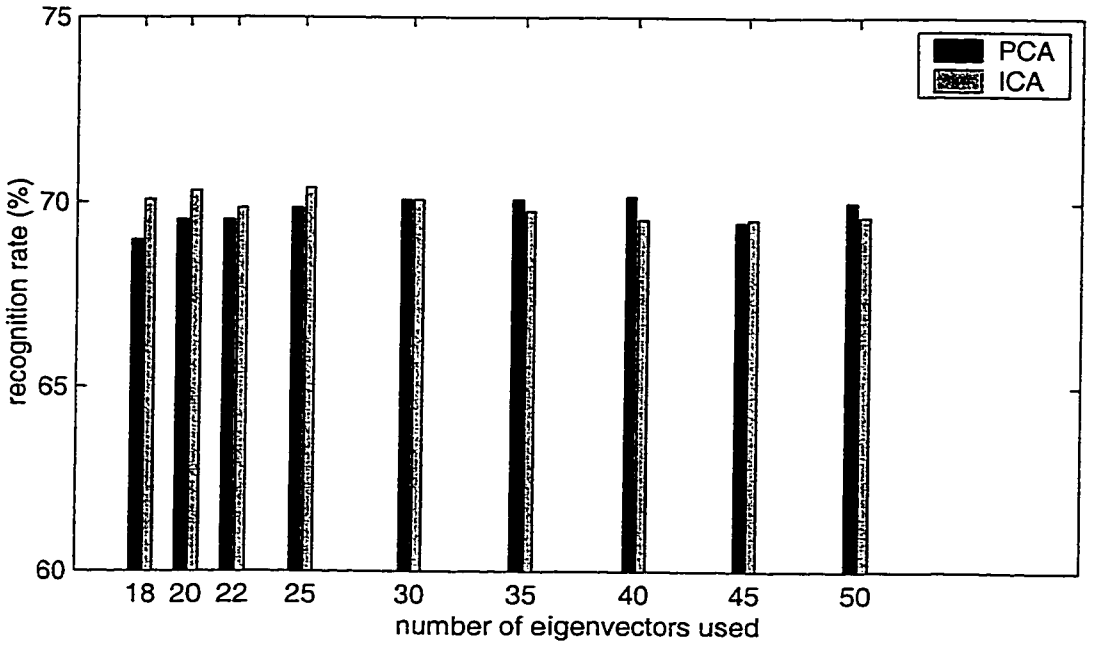


Figure 4.9: Recognition rate of PCA and ICA for training objects sampled at every 50° pose angle.

	number of eigenvectors used								
	18	20	25	30	35	40	45	50	55
ICA	897	900	894	901	897	893	890	890	891
PCA	883	890	890	894	897	897	898	889	896

Table 4.2: The number of correct recognition by using ICA as well as by using PCA. The pose angle sampling was 50°. The recognitions are shown for a total of 1280 test images.

said that the network did not work well if it was forced to extract more number of components than that which actually existed in the data.

4.5.3 Results with pose angle sampling at every 25 degrees

The previous experiment was also repeated by using images sampled at every 25°. This gave a total of 300 training objects, and the rest, 1280 in all, as the test objects. In this case, there was more information for the network to learn. The performance in both the PCA as well as the ICA was increased. The improvement in performance in ICA was again observed. The results are shown in the bar graph in Figure 4.10 and the recognition numbers are shown in Table 4.3. Unlike the case of 50° sampling, the performance of ICA did not go below that of PCA for 50 eigenvectors. Rather, the deterioration in the ICA performance was delayed till the use of around 75 eigenvectors. This could be due to the fact that now more information is available to the network and it needs more number of components to express it. Still, beyond 75 number of independent components, the performance of ICA began to go below that of PCA, again suggesting that it was not advantageous to extract more number of independent components. As the excess number of components being extracted caused the performance to deteriorate, they were considered as noise.

	number of eigenvectors used												
	18	20	25	30	35	40	45	50	55	60	65	70	75
ICA	947	952	950	963	956	958	967	966	961	962	968	963	963
PCA	939	940	944	952	955	951	953	954	960	966	964	961	961

Table 4.3: The number of correct recognition by using ICA as well as by using PCA. The pose angle sampling was 25°. The recognitions are shown for a total of 1280 test images.

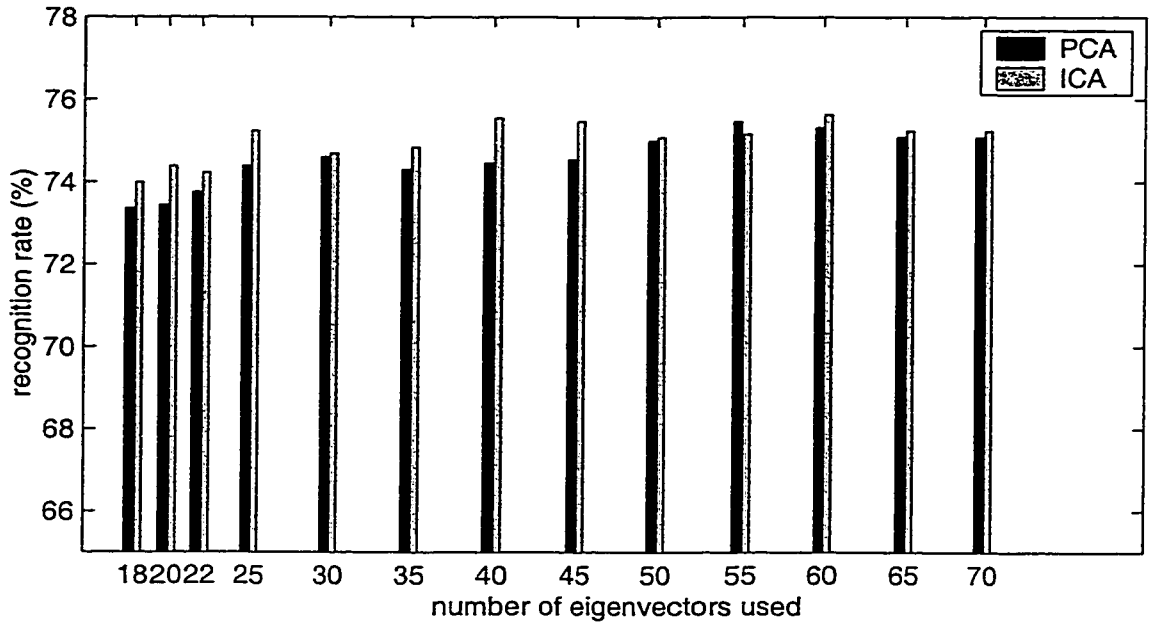


Figure 4.10: Recognition rate of PCA and ICA for training objects sampled at every 25° pose angle. The recognition performance takes into account the object identified as well as the nearest pose in the training data.

4.5.4 Weight initialization and sphering

In the above experiments, the input data was sphered using the whitening filter and then given to the network. The network had its weights initialized to an identity matrix. During training, it was seen that major weight changes took place only in the first few epochs, and after that the weight changes slowed down. They were further slowed down due to annealing. The experiments were also performed under the following different conditions (below, $\mathbf{W}_{Linit}$ is the initialized network weight matrix):

No sphering ($\mathbf{W}_z = \mathbf{I}$), $\mathbf{W}_{Linit} = \mathbf{I}$: This was the case in which the sphering step was avoided but the weight matrix was initialized to unity. The data was made zero mean before being given to the network, though. It was observed that the

weight changes in the first few iterations were, on the average, greater than the case where sphering step was used. This indicated that the network started further away from the convergence point on the entropy surface. The network was trained for the same number of iterations as before. Surprisingly, there was no change in the performance after the first iteration. This suggests that in this particular data set, sphering was not very crucial for the convergence in 1500 iterations.

No sphering ($\mathbf{W}_z = \mathbf{I}$), $\mathbf{W}_{Linit}$ initialized to a random vector: In this case, there was no sphering and the network weight matrix was initialized to elements having a random value between 0 and 1. This case was an interesting one. The performance was severely affected in the first few iterations. The initial average weight change was significantly bigger than the previous cases. For instance, in the case where the training objects were sampled at every 25°, the initial weight change was of the order of 1.5×10^5 while extracting 25 independent components. After a few hundred epochs, the weight change did stabilize, but at a higher value than the previous cases. After the weights stabilized, the recognition performance became same as before.

The above phenomenon stresses the fact that even though sphering was not necessary to improve the recognition performance, it did initialize the neural network in a state favorable to the direction of convergence. In other words, sphering gave the weight change trajectory a head start and the weights reached stable position sooner.

4.6 Database II and results

In the first database, it was observed that the application of ICA gave consistently better performance than PCA. As far as this database is concerned, the results were encouraging. Does this mean that ICA is in general better than PCA? Should

we apply ICA on all vision applications where PCA has been thought of being satisfactory till now? Or, are there situations where ICA may not be a better choice than PCA? To answer these questions more research still needs to be done. But an attempt was made to verify if the improvement in performance of ICA over that of PCA in object recognition could in another database.

It was observed that in the database considered previously, the performance of ICA was consistently better than that of PCA. But consistency by itself does not give the complete picture. The difference in the performance should also be considered before actually making a decision about the superiority of ICA over PCA.

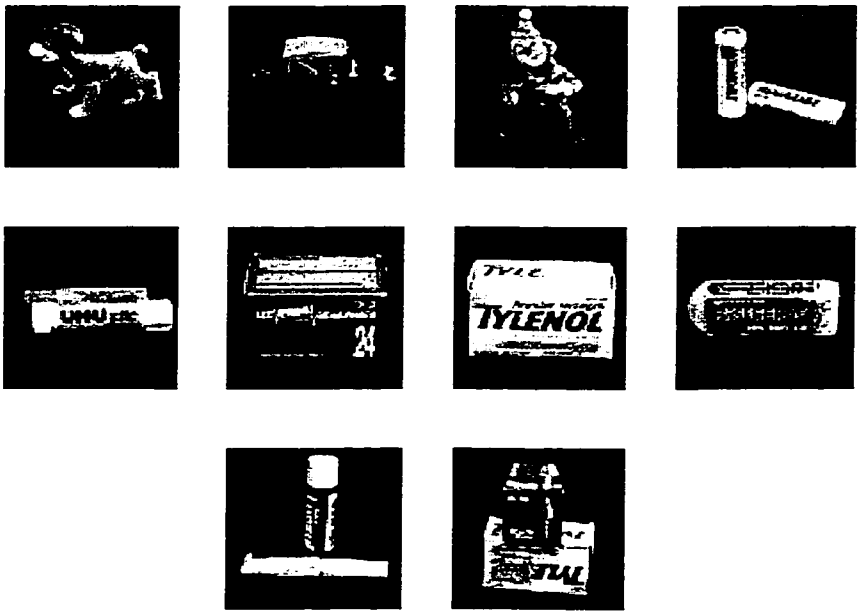


Figure 4.11: The 10 objects database used in the second experiment of 3-D appearance based object recognition.

A new database was constructed of 10 different objects (shown in Figure 4.11). The images of the objects were taken using a KP-M1, a black and white Hitachi Denshi CCD camera with a 16 mm Pentax lens. The images were digitized using the Matrox Meteor-II frame-grabber. Ambient light was used to avoid harsh shadows. The objects were placed on a turntable so that the pose angel could be changed by turning the turntable. The new database, though, was not processed as was the first one. The images were taken with a dark background, but no attempt was made to segment the images and to remove the background, thus giving more realistic images.

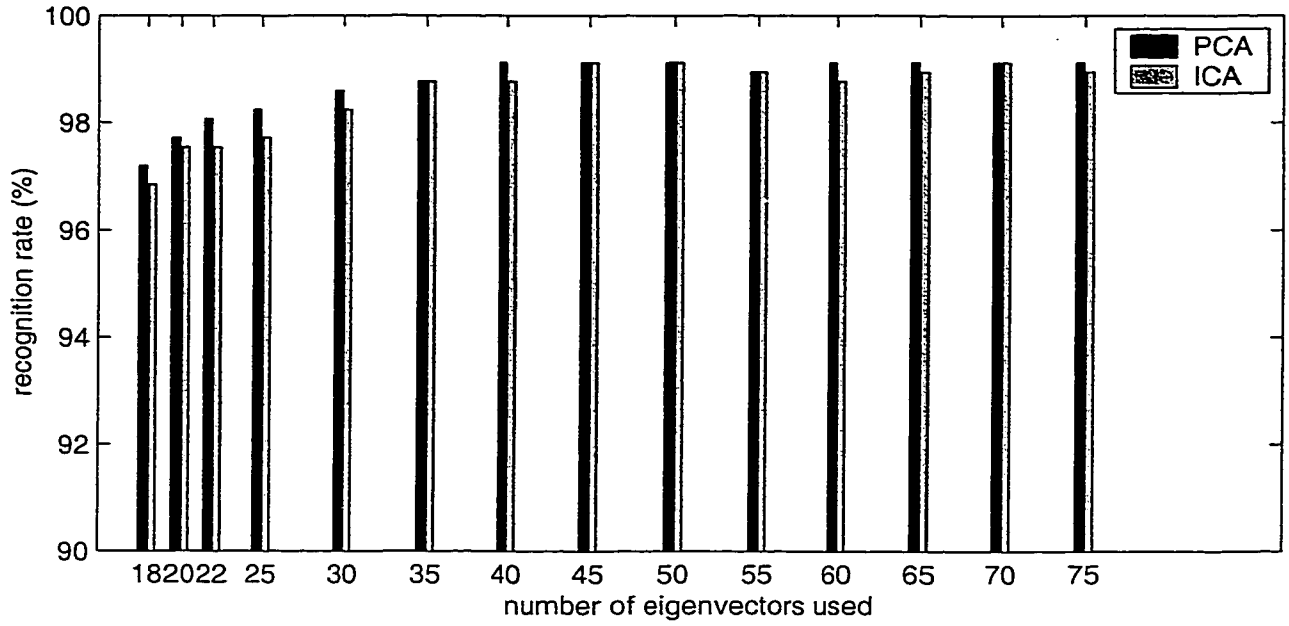


Figure 4.12: Recognition rate of PCA and ICA for training objects in the second database sampled at every 25° pose angle.

The training database was constructed by sampling the pose angle manifold

at 25°. The results are shown in Figure 4.12. It can be seen that the performance of ICA and PCA became similar for more than 55 components being used. It can also be observed that, unlike in the first database, the performance began to stabilize at higher number of components. Based on observations on the second database one can conclude that ICA can not in general be considered as a more powerful tool for object recognition than PCA.

Chapter 5

Conclusions, Discussion and Future Work

5.1 Conclusions and Discussion

In this thesis, ICA was applied for appearance based 3D object recognition. ICA is relatively a new concept, and the results needed to be compared with some other standard transform. PCA was used for comparison purposes since PCA has been widely used in appearance based applications.

Two experiments were performed, one on a pre-processed database in which the object images were segmented from the background and rescaled, and the other on a raw database in which the objects' images were taken against a dark background but not processed as they were in the first case. The training images in both the cases were taken by sampling the pose angle of the objects so that the appearance of the object could be memorized from all angles, thus resulting in appearance based 3D object recognition. Also, PCA was used to pre-process the data in the ICA experiments. This helped to reduce the size of the data on which ICA was performed. The data obtained as the output of PCA was then sphered before being given to the linear feedforward ICA neural network.

In the ICA experiments it was observed that the sphering helped the convergence of the network weights. This could be considered in another way, i.e. sphering gave the weights a head start on the convergence trajectory path. During the learning phase, it was observed that the weights' change was maximum in the first few iterations, and then it gradually became stabilized. It should be noted that stabilization of weights should not be considered as a signal that the network has learned the independent components. Even though the average weights change may not seem significant, the network is still learning; the individual weights changes are significant. So the network was allowed to go on learning even after the weights change stabilized.

In the first database, ICA performed better than PCA in view-invariant appearance based recognition. As the number of components used for the recognition was increased, the performance of ICA gradually started decreasing. This was attributed to the fact that ICA could be trying to extract more number of independent components than were actually present in the data.

In the second case, it was observed that the performance of ICA gradually became the same as that of PCA. So in this case, there was no advantage of using ICA over PCA. This case proved that ICA may not always be suitable for feature extraction. But then it is seldom the case that a transform is a universal transform. Usually a new method is suitable for only a subset of situations. Take the Least Mean Square algorithm for example. Its suitability is dependent on the data, among other things. Specifically, the ratio of the highest eigenvalue of the data to its lowest eigenvalue governs the learning capability of the adaptive filter. Along similar lines, ICA may not be suitable on a particular data set. Since ICA involves higher order statistics, detailed mathematical analysis is not an easy task. Without this, the only way to show non-suitability of ICA is by an example, which was done using the second database.

Hence, it cannot be determined if ICA would be better suited for a recognition application before actually implementing the ICA algorithm and examining its performance. Further research needs to be done in this regards. So far, the most important factor that is usually considered for ICA is the number of independent components to be extracted. In the case of images, this number is never known in advance. This leaves the option of estimating the number of components in some way. In this thesis PCA was used to accomplish this. In most of the natural signals, the number of independent components are always unknown and heuristic or trial and error methods are usually used to make an estimate.

5.2 Scope for Future Work

As the research in ICA progresses, it is being applied to more and more applications. Till the writing of this thesis, we had not come across any other instance of application of ICA to 3D appearance based object recognition. We also believe that the work presented here is only an introduction of such an application of ICA. More work needs to be done before ICA can be declared to be superior to PCA for recognition purposes.

In the databases used, no attempt was made to consider changes in illuminations, occlusion and changes in background. Illumination can tend to change the appearance of an object. Bright changes in illuminations and their effects on ICA performance need to be studied. To determine the differences in performance between ICA and PCA for recognition when the test data images are taken in different illumination settings is a topic of future research. Occlusion has always been a challenging stumbling block in object recognition. Humans seem to deal with occlusion very well, at the same time it is not difficult to observe that this human ability depends a lot on the percentage of occlusion, nature of occlusion, and the context of the occluding object to the occluded object. Even in PCA, it is difficult

to deal with occlusion. To study the effects of occlusion on the performance of ICA can be an interesting research topic, since in realistic conditions, any algorithm has to cope with occlusion at one time or another.

Still other factors that need to be examined regarding ICA for object recognition are the contributions of each extracted independent component. More specifically, if the recognition scenario required that edge information may provide more information, then the set of independent components most useful for recognition may be different than the set most useful when texture information was more advantageous.

Notwithstanding the present complications in the applications of ICA, research in this field is continuing at a fast pace. As the understanding of ICA increases, mathematically as well as experimentally, we firmly believe this new tool will find a foothold in many applications where unsupervised neural networks are needed to learn patterns to extract meaningful information based on higher order statistics.

Bibliography

[1] D. H. Hubel, *Eye, brain, and vision.* New York: Scientific American Library. Distributed by W.H. Freeman, c1988, 1988.

[2] M. D. Levine, *Vision in Man and Machine.* McGraw Hill, 1985.

[3] C. Blakemoreand and F. W. Campbell, "On the existence of neurons in the human visual system selectively sensitive to the orientation and size of retinal images," *Journal of Physiology*, vol. 203, pp. 237–260, 1969.

[4] L. Spillman and J. S. Werner, eds., *Visual Perception, the Neurophysiological Foundation.* Academic Press, 1990.

[5] H. Davson, *Physiology of the Eye.* London: Macmillan Press, 5 ed., 1990.

[6] D. H. Hubel and T. N. Wiesel, "Receptive fields and functional architecture of monkey striate cortex," *Journal of Physiology*, vol. 195, pp. 215–244, 1968.

[7] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *Journal of Physiology*, vol. London, no. 160, pp. 106–154, 1962.

[8] R. T. Chin and C. R. Dyer, "Model-based recognition in robot vision," *ACM Computing Surveys*, vol. 18, no. 1, 1986.

[9] P. J. Besl and R. C. Jain, "Three-dimensional object recognition," *Computing Surveys*, vol. 1, no. 1, pp. 11–23, 1985.

[10] V. S. Nalwa, *A guided tour of computer vision.* Addison-Wesley, 1993.

[11] A. H. Barr, "Superquadratics and angle preserving transformations," *IEEE Computer Graphics and Applications*, vol. 1, pp. 11–23, January 1981.

[12] D. Keren, D. Cooper, and J. Subrahmonia, "Describing complicated objects by implicit polynomials," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, pp. 38–53, January 1994.

[13] Y.-H. Tseng, J.-N. Hwang, and F. H. Sheeham, "Three-dimensional object representation and invariant recognition using continuous distance transform neural networks," *IEEE Transactions on Neural Networks*, vol. 8, pp. 141–147, January 1997.

[14] L. E. Weiss, A. Sanderson, and C. P. Neuman, "Dynamic sensor-based control of robots with visual feedback," *IEEE Transactions on Robotics and Automation*, vol. RA-3, pp. 404–417, October 1987.

[15] J. T. Feddema, C. S. G. Lee, and O. R. Mitchell, "Weighted selection of image features for resolved rate visual feedback control," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 1, pp. 31–47, 1991.

[16] T. Poggio and S. Edelman, "A network that learns to recognize 3d objects," *Nature*, vol. 343, pp. 263–266, 1990.

[17] J. Weng, N. Ahuja, and T. S. Huang, "Learning recognition and segmentation of 3-d objects from 2-d images," in *Proceedings of Fourth Int'l Conf. on Computer Vision*, (Berlin), pp. 121–128, May 1993.

[18] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proceedings IEEE Computer Society Conference Computer Vision and Pattern Recognition*, (Maui. Hawaii), pp. 586–591, 1991.

[19] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, vol. 3, no. 1, 1991.

[20] H. Murase and S. K. Nayar, "Visual learning and recognition of 3-d objects from appearance," *International Journal of Computer Vision*, vol. 14, no. 1, pp. 5–24, 1995.

[21] S. K. Nayar, S. A. Nene, and H. Murase, "Subspace methods for robot vision," *IEEE Transactions on Robotics and Automation*, vol. 12, p. 750, October 1996.

[22] L. Sirovich and M. Kirby, "Low-dimensional procedure for the characterization of human faces," *Journal of the Optical Society of America*, vol. 4, no. 3, pp. 519–524, 1987.

[23] A. Pentland, B. Moghaddam, and T. Starner, "View-based and modular eigenspaces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 84–91, 1994.

[24] E. Oja, *Subspace Methods of Pattern Recognition.* Research Studies Press, England: Letchworth, 1983.

[25] K. Fukunaga, *Introduction to Statistical Pattern Recognition.* London: Academic Press, 1990.

[26] H. Murase and S. K. Nayar, "Learning and recognition of 3d objects from appearance," in *IEEE 2nd Qualitative Vision Workshop*, (New York), 1993.

[27] A. R. Pope and D. G. Lowe, "Learning object recognition models from images," in *Proc. of Fourth International Conference on Computer Vision*, (Berlin), pp. 296–301, May 1993.

[28] D. P. Huttenlocher, R. H. Lilien, and C. F. Olson, "View-based recognition using an eigenspace approximation to the hausdorff measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, p. 951, September 1999.

[29] J. J. Atick and A. N. Redlich, "Towards a theory of early visual processing," *Neural Computation*, vol. 2, pp. 308–320, 1990.

[30] S. Haykin, *Neural Networks A Comprehensive Foundation*. New Jersey: Prentice Hall, second ed., 1999.

[31] F. Attneave, "Some informational aspects of visual perception," *Psychological Review*, vol. 61, pp. 183–193, 1954.

[32] H. B. Barlow, *Sensory Communication*, ch. Possible principles underlying the transformations of sensory messages, pp. 217–234. MIT Press, 1961.

[33] H. B. Barlow, *Vision: Coding and Efficiency*, ch. A theory about the functional role and synaptic mechanism of visual after-effect. Cambridge University Press, 1987.

[34] H. B. Barlow, "Unsupervised learning," *Neural Computation*, vol. 1, pp. 295–331, 1989.

[35] J. J. Atick, "Could information theory provide an ecological theory of sensory processing?," *Network*, vol. 3, pp. 213–251, 1992.

[36] J. J. Atick and A. N. Redlich, "What does the retina know about natural scenes," *Neural Computation*, vol. 4, pp. 196–210, 1992.

[37] Y. Dan, J. J. Atick, and R. C. Reid, "Efficient coding of natural scenes in the lateral geniculate nucleus: Experimental test of a computational theory," *Journal of Neuroscience*, vol. 16, pp. 1351–1362, 1996.

[38] D. W. Dong and J. J. Atick, "Temporal decorrelation - a theory of lagged and nonlagged responses in the lateralgeniculate-nucleus," *Network Computation in Neural Systems*, vol. 6, no. 2, pp. 159–178, 1995.

[39] D. J. Field, "Scale-invariance and self-similar 'wavelet' transforms: an analysis of natural scenes and mammalian visual system," in *Wavelets, Fractals, and Fourier Transforms* (M. Farge, J. Hunt, and C. Vascillicos, eds.), pp. 151–193, 1993.

[40] B. A. Olshausen and D. J. Field, "Natural image statistics and efficient coding," *Network*, no. 7, pp. 333–339, 1996.

[41] J. M. Zurada, *Introduction to artificial neural systems.* New York: St. Paul, 1992.

[42] Y.-T. Zhou, *Artificial neural networks for computer vision.* Springer-Verlag, 1992.

[43] J. Zhang, Y. Yan, and M. Lades, "Face recognition: Eigenface, elastic matching, and neural nets," *Proceedings of the IEEE*, vol. 85, pp. 1423–1435, September 1997.

[44] R. C. Gonzalez and R. C.Woods, *Digital image processing.* Reading, Mass.: Addison-Wesley, 1992.

[45] H. Bourlard and Y. Kamp, "Auto-association by multilayer perceptrons and singular value decomposition," *Biological Cybernetics*, vol. 59, pp. 291–294, 1988.

[46] D. Hebb, *The Organization of Behavior.* New York: Wile, 1949.

[47] E. Oja, "A simplified neuron model as a principal component analyzer," *Journal of Mathematical Biology*, vol. 15, 1982.

[48] T. D. Sanger, "Optimal unsupervised learning in a single-layer feedforward neural network," *Neural Networks*, vol. 2, pp. 459–473, 1989.

[49] M. Abramowitz and I. A. Stegun, eds., *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, 9th printing*, ch. Probability Functions. United States Department of Commerce, 1970.

[50] T. M. Cover and T. A. Thomas, *Elements of Information Theory*. New York: John Wiley and Sons, 1991.

[51] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 1948.

[52] *A mathematical theory of communication*. University of Illinois Press, 1963.

[53] R. Linsker, "Self-organization in a perceptual network," *Computer*, vol. 21, pp. 105–117, 1988.

[54] R. Linsker, *Neural Information Processing Systems*, ch. Towards an organizing principle for a layered perceptual network, pp. 485–494. American Institute of Physics, 1988.

[55] S. Kullback, *Information theory and statistics*. New York: Dover Publications, 1968. An unabridged republication of the work originally published in 1959.

[56] J. E. Shore and R. W. Johnson, "Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy," *IEEE Transactions on Information Theory*, vol. IT-26, pp. 26–37, 1980.

[57] P. Comon, "Independent component analysis - a new concept?," *Signal Processing*, vol. 36, no. 3, pp. 287–314, 1994.

[58] C. Jutten and J. Herault, "Blind separation of sources, part 1: An adaptive algorithm based on neuromimetic architecture," *Signal Processing*, vol. 24, no. 1, pp. 1–10, 1991.

[59] A. Bell and T. Sejnowski, "The independent components of natural scenes are edge filters," *Vision Research*, vol. 37, no. 23, pp. 3327–3338, 1997.

[60] N. Delfosse and P. Loubaton, "Adaptive blind separation of independent sources : a deflation approach," *Signal Processing*, vol. 45, pp. 59–83, 1995.

[61] J. F. Cardoso and B. H. Laheld, "Equivariant adaptive source separation," *IEEE Trans. on Signal Processing*, vol. 44, no. 12, pp. 3017–3030, 1996.

[62] A. Cichocki and R. Unbehauen, *Neural Networks for Signal Processing and Optimization.* Wiley, 1994.

[63] A. Cichocki and R. Unbehauen, "Robust neural networks with on-line learning for blind identification and blind separation of sources," *IEEE Trans. on Circuits and Systems*, vol. 43, no. 11, pp. 894–906, 1996.

[64] R. H. Lambert, *Multichannel Blind Deconvolution: FIR Matrix Algebra and Separation of Multipath Mixtures.* PhD thesis, University of Southern California, 1996.

[65] E. Sorouchyari, "Blind separation of sources, part iii," *Signal Processing*, vol. 24, pp. 21–29, 1991.

[66] M. Lindgren, T. Wigren, and H. Broman, "On local convergence of a class of blind separation algorithms.," *IEEE Trans. on Signal Processing*, vol. 43, pp. 3054–3058, 1995.

[67] A. Cichocki, R. Unbehauen, L. Moszczynski, and E. Rummert, "A new on-line adaptive algorithm for blind separation of source signals.," in *Proceedings of*

International Symposium on Artificial Neural Networks ISANN-94, (Taiwan), pp. 406–411, 1994.

[68] A. Hyvärinen and E. Oja, "Simple neuron models for independent component analysis," *International Journal of Neural Systems*, 1997.

[69] A. Bell and T. Sejnowski, "An information maximization approach to blind separation and blind deconvolution," *Neural Computation*, vol. 7, pp. 1129–1159, 1995.

[70] S. Amari, A. Cichocki, and H. H. Yang, "A new learning algorithm for blind signal separation," in *Advances in Neural Information Processing Systems 8* (D. Touretzky, M. Mozer, and M. Hasselmo, eds.), pp. 752–763, Cambridge MA: MIT Press, 1996.

[71] S. Amari, "Natural gradient works efficiently in learning," *Neural Computation*, vol. 10, pp. 251–276, 1998.

[72] S. Amari, S. C. Douglas, A. Cichocki, and H.H.Yang, "Multichannel blind deconvolution and equalization using the natural gradient," in *Proc. of IEEE International Workshop on Wireless Communication*, pp. 101–104, April 1997.

[73] J.-P. Nadal and N. Parga, "Non-linear neurons in the low-noise limit: a factorial code maximizes information transfer," *Network*, vol. 4, pp. 295–312, 1994.

[74] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*. McGraw Hill, 1991.

[75] G. Strang, *Introduction to linear algebra*. Wellesley, MA: Wellesley-Cambridge Press, 1993.

[76] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library (coil-20)," Technical Report CUCS-005-96, Columbia University, February 1996. The database is available at http://www.cs.columbia.edu/CAVE/.