KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY

Digital Image Processing Via Singular Value Decomposition

By

Gideon Kwadzo Gogovi

BSc.(Mathematics)

A Thesis submitted to the Department of Mathematics, Kwame Nkrumah University of Science and Technology in partial fulfillment of the requirements for the degree of

Master of Philosophy

College of Science

MAY 2013

Declaration

I hereby declare that this submission is my own work towards the award of the M.Phil degree and that, to the best of my knowledge, it contains no material previously published by another person nor material which had been accepted for the award of any other degree of the university, except where due acknowledgement had been made in the text.

Gideon Kwadzo Gogovi	....	....
(Student)	Signature	Date
Certified by:		
Dr. F. T. Oduro	....	....
(Supervisor)	Signature	Date
Certified by:		
Mr. F.K Darkwa	....	....
(Head of Department)	Signature	Date

Acknowledgements

To God, the Father, the giver and maker of vision. Thank you for a dream come true.
I would like to express my heartfelt appreciation and gratitude to my supervisor Dr. F.T. Oduro for his support. The inexhaustible supply of help, advice and most importantly patience has been invaluable in the completion of this thesis. I would also like to thank Prof. Francis Benyah of the University of the Western Cape, South Africa for introducing me to the concept of Singular Value Decomposition. In addition, I wish to thank my coursemates who provided many light-hearted moments during our time together. I should especially acknowledge all the lecturers of the Department of Mathematics and Statistics, University of Cape Coast, who have been a fountain of knowledge and great friends throughout the years.
Finally, I would like to thank my uncle Mr. Walter C.K Dogli for all of the time, effort and money he spent on getting my dream in education come true. Without his help I would not have managed to get to where I am today. God bless you Mr. Walter C.K Dogli.

Dedication

To my uncle Mr. Walter C.K Dogli, my mother Gladys Kutiame and my late grandmother Awusi Klokpa

Abstract

In this thesis, well studied linear algebra theory "singular value decomposition" (SVD) and its applications is presented. Singular Value Decomposition is extraordinarily useful and has many applications such as data analysis, signal processing, pattern recognition, objects detection and weather prediction. SVD method can transform matrix A into product USV^T. Some of these application areas discussed include the Moore-Penrose psuedoinverse, the low rank approximation of matrices, the least square solution to linear systems and image face recognition. To perform face recognition with SVD, the set of known faces were treated as vectors in a subspace, called "face space", spanned by a small group of "basefaces". The projection of a new image onto the baseface was then compared to the set of known faces to identify the face. The study also investigated the characteristics of singular values and singular vectors in image processing. SVD was found to be a stable and effective method to decompose a system into a set of linearly independent components. The approach is robust, simple, easy and fast to implement and provides a practical solution to image recognition problem. It was also found that, though the singular values are unique, the singular vectors are more important in image processing. MATLAB R2012a with image processing toolbox was used as the development tool for implementing the algorithm.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This chapter introduced the history and concept of Singular Value Decomposition and gave a brief definition of an image. It also discussed the objectives of the study. Finally, the organization of the thesis was done.

1.1 Background of the study

Singular value decomposition (SVD) is an important concept in linear algebra. Consequently relatively few mathematicians are familiar with what the Massachusetts Institute of Technology Professor, Gilbert Strang calls “absolutely a high point of linear algebra”(Autonne, 1913). According to (Soumya, Soumya, & Soman, 2009) Singular Value Decomposition was developed in the mid 19^{th} century but most of its applications emerged in the 21st century. Singular Value Decomposition (SVD) for square matrices was developed by Eugenio Beltrami (1873), Camille Jordan (1874), James Joseph Sylvester (1889) and Autonne (1915). Eckart and Young developed Singular Value Decomposition in the 1930’s for rectangular matrices and its use as a computational tool dates back to the 1960’s. Singular values and the singular value decomposition play an important role in high-quality statistical computations and in schemes for data compression based on approximating a given matrix with one of lower rank. They also play a central role in the theory of unitarily invariant norms. Many modern computational algorithms are based on singular value computations because the problem of computing the eigenvalues of a general matrix (like the problem of computing the eigenvalues of a Hermitian matrix) is well-conditioned.

The singular value decomposition (SVD) is known by many names, such as principal component analysis. In numerical analysis, the singular value decomposition provides a measure of the effective rank of a given matrix. In statistics and time series analysis, the singular value decomposition is a particularly useful tool for finding least-squares solutions and approximations.

Singular Value Decomposition is a widely used technique to decompose a matrix into several component matrices, exposing many of the useful and interesting properties of the original matrix. The decomposition of a matrix is often called a factorization. Ideally, the matrix is decomposed into a set of factors (often orthogonal or independent) that are optimal based on some criterion. The decomposition of a matrix is also useful when the matrix is not of full rank. That is, the rows or columns of the matrix are linearly dependent. Theoretically, one can use Gaussian elimination to reduce the matrix to row echelon form and then count the number of nonzero rows to determine the rank. However, this approach is not practical when working in finite precision arithmetic. A similar case presents itself when using LU decomposition where L is in lower triangular form with 1's on the diagonal and U is in upper triangular form.

Ideally, a rank-deficient matrix may be decomposed into a smaller number of factors than the original matrix and still preserve all of the information in the matrix. The singular value decomposition, in general, represents an expansion of the original data in a coordinate system where the covariance matrix is diagonal. Using the singular value decomposition, one can determine the dimension of the matrix range or more-often called the rank. The rank of a matrix is equal to the number of linear independent rows or columns. This is often referred to as a minimum spanning set or simply a basis. The singular value decomposition can also quantify the sensitivity of a linear system to numerical error or obtain a matrix inverse. Additionally, it provides solutions to least-squares problems and handles situations when matrices are either singular or numerically very close to singular. In singular value decomposition transformation, a matrix will be decomposed into three matrices that are of the same size as the original matrix.

Images and Digital Images

From the view point of linear algebra, an image is an array of non-negative scalar entries, or a matrix of square pixels (picture elements) arranged in columns and rows. A digital image differs from a photo in that the values are all discrete. Usually they take on only integer values. A digital image can be considered as a large array of discrete dots, each of which has a brightness associated with it. These dots are called picture elements, or more simply pixels. The pixels surrounding a given pixel constitute its neighborhood. A neighborhood can be characterized by its shape in the same way as a matrix. According to Robert M. Gray and David L. Neuhoff, a digital image is a discrete two-dimensional function, $f(x, y)$ which has been quantized over its domain and range (Gray & Neuhoff., 1998). Without loss of generality, it will be assumed that the image is rectangular, consisting of Y rows and X columns. The resolution of such an image is written as $X \times Y$. By convention, $f(0, 0)$ is taken to be the top left corner of the image, and $f(X - 1, Y - 1)$ the bottom right corner. The distinct coordinates in this image is the pixel. The nature of the output of $f(x, y)$ for each pixel is dependent on the type of image. Most images are the result of measuring a specific physical phenomenon, such as light, heat, distance, or energy. The measurement could take any numerical form. This is summarized in the figure below.

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & \cdots & f(0,Y-1) \\ f(1,0) & f(1,1) & f(1,2) & \cdots & f(1,Y-1) \\ f(2,0) & f(2,1) & f(2,2) & \cdots & f(2,Y-1) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ f(X-1,0) & f(X-1,1) & f(X-1,2) & \cdots & f(X-1,Y-1) \end{bmatrix}$$

Figure 1.1: A rectangular digital image.

Some aspects of image processing different face recognition from image compression includes image enhancement, Image Restoration and Image Segmentation. Processing an image so that the result is more suitable for a particular application is referred to as image enhancement. This deals with sharpening or deblurring an out of focus image, highlighting edges, improving image contrast, or brightening an image, removing noise (any degradation in the image signal, caused by external disturbance). Image restoration may be considered as reversing the damage done to an image by a known cause. This include removing of blur caused by linear motion and the removal of optical distortions. Image segmentation involves subdividing an image into constituent parts, or isolating certain aspects of an image. This is normally done by finding lines, circles, or particular shapes in an image in an aerial photograph, identifying cars, trees, buildings, or roads. People use digital images in many ways. The same image can be viewed on a wide variety of monitors, printed in many formats, and transmitted electronically through electronic-mails, cell phones, and other systems. Digital images are stored electronically on media such as computer hard drives, CDs, DVDs and magnetic tapes.

Types of Digital Images

The various types of digital image include the Binary, Grayscale and the True Color or RGB images.

Binary image:

In the binary image, each pixel is just black or white. Since there are only two possible values for each pixel (0,1), we only need one bit per pixel. Such images can therefore be very efficient in terms of storage. Images for which a binary representation may be suitable include text (printed or handwriting), fingerprints, or architectural plans. An example of a binary image is shown in the figure(1.2).

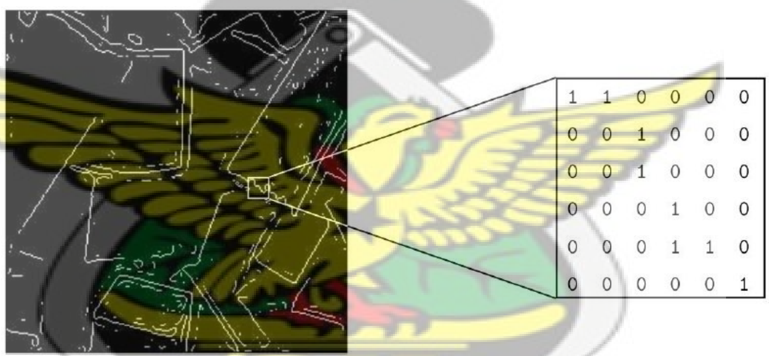

Figure 1.2: An Example of a Binary Image

Grayscale images:

In Grayscale images, each pixel is a shade of gray, normally from 0 (black) to 255 (white). This range means that each pixel can be represented by eight bits, or exactly one byte. Other grayscale ranges are used but generally they are a power of 2. Such images arise in medicine (X-rays), images of printed works, and indeed different gray levels is sufficient for the recognition of most natural objects. An example of a grayscale image is shown in the figure(1.3).

Figure 1.3: An Example of Grayscale Image

True Color or RGB

In True Color or RGB each pixel has a particular color; that color is described by the amount of red, green and blue in it. If each of these components has a range 0 - 255, this gives a total of $255^3 = 16,581,375$ different possible colors. Such an image is a "stack" of three matrices; representing the red, green and blue values for each pixel. This means that for every pixel, there corresponds 3 values. An example of a true color or RGB image is shown in the figure (1.4).

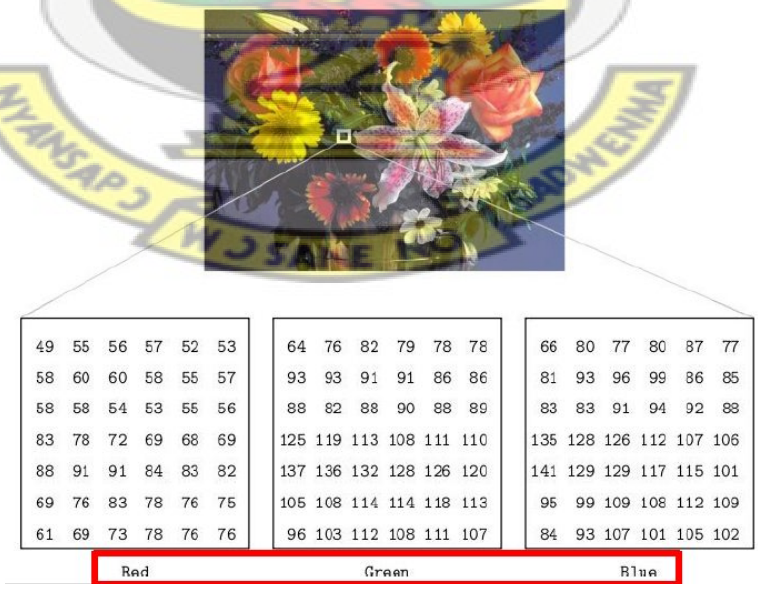

Figure 1.4: An Example of a True Color or RGB Image

Spatial Resolution

Spatial resolution is the density of pixels over the image. The greater the spatial resolution, the more pixels are used to display the image. Halving the size of the image, by taking out every other row and every other column, that is, leaving only those matrix elements whose row and column indices are even. Doubling the size of the image, all the pixels are repeated to produce an image with the same size as the original, but with half the resolution in each direction.

1.2 Statement Of Problem

The rapid evolution of digital technology has improved the ease of access to digital information enabling reliable, faster and efficient storage, transfer and processing of digital data. The need for storage spaces has increased the rate at which newer versions of personal computers are produced. The issue of storage space brought about the idea of producing newer versions of personal computers which has made the field of image processing a very important area to research. Instead of producing storage devices with higher storage capacities, image processing (compression) based on singular value decomposition, rather reduces the size of an image(data) to enable smaller storage space to be used.

The Singular Value Decomposition (SVD) is a popular matrix factorization that has been used widely in applications ever since an efficient algorithm for its computation was developed in the 1970s (Stewart, 1992). In recent years, the Singular Value Decomposition has become even more prominent due to a surge in applications and increased computational memory and speed.

Humans have evolved very precise visual skills such that we can identify a face in an instant, differentiate colours and process a large amount of visual information very quickly. But the problem is that, the world is in constant motion. As one stares at something long enough and it will change in some way. Even a large solid structure, like a building or a mountain, will change its appearance depending on the time of day (day or night), amount of sunlight (clear or cloudy) or various shadows falling upon it. This narrows down to the nucleus of the study, the problem of face recognition.

Face recognition is one of the most relevant applications of image analysis. It is a true challenge to build an automated system which equals human ability to recognize faces. Although humans are quite good in identifying known faces, we are not very skilled when we must deal with a large amount of unknown faces. The computers, with an almost limitless memory and computational speed, should overcome such humans limitations.

1.3 Research Objectives

1.3.1 General Objective

The main objective of this study was to process digital images using the concept of singular value decomposition(SVD).

1.3.2 Specific Objectives

The specific objectives of this study are as follows:

- To review the concept of singular value decomposition.
- To investigate the characteristics of singular values and singular vectors in image processing.

- To develop a program for face recognition using singular value decomposition.

1.4 Outline of the study

This thesis contains five main chapters. In chapter one, the concept of Singular Value Decomposition and image processing were introduced. In Chapter two, a comprehensive theoretical framework, review of the related literature in the field which served as the basis for the theoretical framework for the study was discussed. Chapter three presented a detailed outline of the Singular Value Decomposition procedure and its application to digital image processing. In Chapter four, interpretive critique and discussion of the results of the study are presented. Chapter five concludes the entire study by stating the way for further study based on the major findings made in the study.

Chapter 2

Review of Literature

2.1 Introduction

In this chapter, some basic concepts and notations relating to the concept of singular value decomposition are discussed. This chapter also reviewed literature in the area of Singular Value Decomposition and image processing. Major theories, arguments, methodologies, approaches and controversies in the existing literature on the subject of this study are discussed in this chapter. Some of the applications of the singular value decomposition as a method in linear algebra are also discussed.

2.2 Some basic definitions and Notations

Matrix Norms

A general matrix norm is a function $\| \cdot \|$ from the set of all complex matrices (of all finite orders) into $\Re$ that satisfies the following properties.

$\| A \| \geq 0$ and $\| A \| = 0 \Leftrightarrow A = 0$.

$\| \alpha A \| = |\alpha| \, \| A \|$ for all scalars α

$\| A + B \| \leq \| A \| + \| B \|$ for matrices of the same size.

$\| AB \| \leq \| A \| \| B \|$ for all conformable matrices.

The Frobinius Matrix Norm

The space $C^{m\times n}$ is a vector space of dimension $m \times n$, magnitude of matrices $A \in C^{m\times n}$ can be "measured" by employing any vector norm on $C^{m\times n}$. This called the Frobinius norm.

The Frobinius norm of $A \in C^{m\times n}$ is defined by the equations

$$\| A \|_F^2 = \sum_i^m \sum_j^n |a_{ij}|^2 = \sum_i \| A_{i*} \|_2^2 = \sum_j \| A_{*j} \|_2^2 = trace(A^T A)$$

The Frobinius norm, which satisfies the above definition of matrix norm, is fine for some problems, but it is not well suited for all applications.

Induced Matix norm

A vector norm that is define on $C^{m\times n}$ induces a matrix norm on $C^{m\times n}$ by setting

$$\| A \| = \max_{\|x\|=1} \| Ax \|$$

for $A \in C^{m\times n}$, $x \in C^{n\times 1}$. The induced norm is compatible with its underlying vector norm in the sense that $\| Ax \| \leq \| A \| \| x \|$.

When A is singular,

$$\min_{\|x\|=1} \| Ax \| = \frac{1}{\| A^{-1} \|}$$

The 2-norm of a matrix A is the square root of the largest eigenvalue of $A^T A$.

That is

$$\| A \|_2 = \sqrt{\max_{\mu \in \lambda(A^T A)} \mu}$$

A square matrix A is said to be orthogonal if $A^T A = I$. It is well known that if A is orthogonal, then $\| Ax \|_2 = \| x \|_2$.

The column rank and the row rank of a matrix are defined as the dimensions of its column space and row space, respectively. In image processing (compression), an application of Singular Value Decomposition, if the matrices $A, A_k \in \Re^{m\times n}$ represent an original image and its compressed version respectively, and rank $(A_k) = k$, then the compression ratio of A_k with respect to A is given by $(m+n)k/mn$.(Pagadala, 1998)

2.3 History of the Singular Value Decomposition

The singular value decomposition(SVD) has a long history. It was originally developed in the Nineteenth-century by differential geometers and algebraists who wanted to determine, for given matrices $A = [a_{ij}]$ and $B = [b_{ij} \in M_n(R)]$, whether the two bilinear forms: $\Phi_A(x, y) = \sum_{i,j=1}^{n} a_{ij}x_iy_i$ and $\Phi_B(x, y) = \sum_{i,j=1}^{n} b_{ij}x_iy_i$ could be made equal for $x = [x_i]$ and $y = [y_i] \in R^n$, under independent real orthogonal transformation of spaces it acts on. That is, does there exist $Q_1, Q_2 \in M_n(R)$ such that $\Phi_B(Q_1x, Q_2y) = \Phi_A(x, y)$ for all $x, y \in R^n$?

This problem could be approached by finding a canonical form to which any such bilinear form can be reduced by orthogonal substitution, or by finding a complete set of invariants for a bilinear form under orthogonal substitutions. This is the original motivation for the study of singular values. Though the singular value decomposition is over 100 years old, it became popular only by the work of Gene

Golub and numerical analyst such as Van Loan in the late 1960's. Gene Golub and Van Loan demonstrated its usefulness and feasibility in a wide variety of applications.

Actually in 1873, Beltrami (Stewart, 1992) discovered the singular value decomposition of matrices. Beltrami's contribution appeared in the journal of mathematics for the use by Italian Universities. He established an algorithm to determine the diagonalising transformation. Beltrami discovered the singular value decomposition for real, square, non-singular matrices having distinct singular values. But it lacks the extras needed to handle degeneracies (Beltrami, 1990). Beltrami began with a bilinear form $f(x, y) = x^T Ay$, where A is real, nonsingular and of order n. Making the substitution $x = U\xi$ and $y = V\eta$, then $f(x, y) = \xi^T S\eta$ where

$$S = U^T AV. \tag{2.1}$$

Beltrami then observed that if U and V are required to be orthogonal, then there are $n^2 - n$ degrees of freedom in their choice, and he proposes to use these degrees of freedom to annihilate the off diagonal elements of S. Assuming S is diagonal; that is $S = diag(\sigma_1, \sigma_2, ..., \sigma_n)$. Then it follows from the orthogonality of V that, $U^T = SV^T$ and similarly, $AV = US$

Substituting the value of U obtain from $AV = US$ into $U^T A = SV^T$, Beltrami obtained the equation

$$U^T(AA^T) = S^2 U^T, \tag{2.2}$$

And respectively he obtained $(A^T A)V = VS^2$.

Therefore, the singular values are the roots of the equations $det(AA^T - \sigma^2 I) = 0$ and $det(A^T A - \sigma^2 I) = 0$

Beltrami next stated that by a well-known theorem, these singular values are real. Moreover, they are positive. This follows from theory of quadratic forms. That is

$$0 < \| x^T A \|^2 = x^T(AA^T)x = \xi^T S^2 \xi \tag{2.3}$$

There is some confusion here. Beltrami appears to be assuming the existence of the vector ξ, whose very existence he is trying to establish. The vector required by his argument is an eigenvector of AA^T corresponding to σ. But the fact that the two vectors turn out to be the same caused him to apparently leap ahead of his assumption to use ξ equation above.

Independently, in 1874, the French algebraist Camille Jordan, can fairly be called the co-discoverer of the singular value decomposition (Stewart, 1992). Jordan treated three problems, of which the reduction of a bilinear form to a diagonal form by orthogonal substitution is the simplest. He started with the form $P = x^T Ay$ and seeks to maximize and minimize P subject to $\| x \|^2 = \| y \|^2 = 1$. The maximum is determine by the equation

$$0 = dP = dx^T Ay + x^T Ady, \tag{2.4}$$

which must be satisfied for all dx and dy that satisfy $dx^T x = 0$ and $dy^T y = 0$. Jordan then stated categorically that, the equation $dP = 0$ will therefore be the combination of $dx^T x = 0$ and $dy^T y = 0$.

That is;

$$dx^T A + x^T Ady = dx^T x + dy^T y$$

$$dx^T Ay = dx^T x$$

$$Ay = \sigma x \tag{2.5}$$

$$x^T(Ay) = \sigma x^T x = \sigma$$

Similarly,

$$x^T A dy = dy^T y$$

$$x^T A = \rho y^T \tag{2.6}$$

$$y(x^T A) = \rho y^T y = \rho$$

It follows that the maximum is

$$x^T(Ay) = y(x^T A) = \sigma x^T x = \rho y^T y = \sigma = \rho \tag{2.7}$$

According to Jordan, the σ is determined by the vanishing of the determinant

$$D = \begin{vmatrix} -\sigma I & A \\ A^T & -\sigma I \end{vmatrix}$$

which contains only even powers of σ.

Now let σ_1 be a root of the equation $D = 0$, and let the equations (2.5) and (2.6) be satisfied by $x = u$ and $y = v$ where $\| u \|^2 = \| v \|^2 = 1$. Let $\hat{U} = (uU_*)$ and $\hat{V} = (vU_*)$ be orthogonal, and let $x = \hat{U}\hat{x}$ and $y = \hat{V}\hat{y}$. With these substitutions, let $P = \hat{x}^T \hat{A}\hat{y}$. In this system, P attains its maximum for $\hat{x} = \hat{y} = e_1$ where $e_1 = (1, 0, ..., 0)^T$.

Moreover, at the maximum we have $\hat{A}\hat{y} = \sigma_1 \hat{x}$ and $\hat{x}^T \hat{A} = \sigma_1 \hat{y}^T$ which implies that

$$\hat{A} = \begin{bmatrix} \sigma & 0 \\ 0 & A_1 \end{bmatrix}$$

Thus with $\xi_1 = \hat{x}_1$ and $\eta_1 = \hat{y}_1$, P assumes the form $\sigma_1 \xi_1 \eta_1 + P_1$ where P_1 is independent of ξ_1 and η_1. Jordan now applies the reduction inductively to P_1 to arrive at the canonical form

$$P = \xi^T \Sigma \eta$$

Finally, Jordan notes that when the roots of the characteristic equation $D = 0$ are simple, the columns of U and V can be calculated directly from equations (2.5) and (2.6).

Jordan's approach of using a partial solution of the problem to reduce it to one of smaller size deflation is the modern term which avoids the degeneracies that complicated Beltrami's approach. Incidentally, the technique of deflation apparently lay fallow until Schur, according to (Stewart, 1992) used it to establish his triangular form of a general matrix. It is now a widely used theoretical and algorithmic tool. Another consequence of Jordan's approach is the variational characterization of the largest singular values as the maximum of a function.

James Joseph Sylvester wrote a footnote and two papers on the subject of singular value decomposition. The footnote appears at the end of "The Messenger of Mathematics" (Sylvester, 1889a) entitled as "A new proof that a general quadratic may be reduced to its canonical form (That Is, a Linear Function of Squares) by means of real orthogonal substitutions". In his paper he described the iterative algorithm for reducing a quadratic form to a diagonal form. Sylvester, Jacobi and Beltrami used linear algebra to solve the singular value decomposition. Sylvester describes an iterative algorithm for reducing a quadratic form to orthogonal form. He presented this algorithm in a final messenger paper.(Sylvester, 1889b).

Sylvester also began with a bilinear form $B = x^T A y$ and considered the quadratic form

$$M = \sum_i (\frac{dB}{dy_i})^2$$

Let $M = \sum \lambda_i \xi_i^2$ be the canonical form of M. If B has the canonical form $B = \sum \sigma_i \xi_i \eta_i$ then $\sum [\sigma_i \xi_i]^2$ is orthogonally equivalent to $M = \sum \lambda_i \xi_i^2$ which implies that $\lambda_i = \sigma_i^2$ in some order.

Sylvester introduced the matrix $M = AA^T$ and $N = A^T A$ to find the substitution and stated categorically that the substitution for x is the substitution that diagonalizes M and the substitution for y is the one that diagonalizes N. Meanwhile, this is true only if the singular values of A are distinct.

According to Sylvester(1889b), the following rule can be used to find the coefficients of the x-substitution conrresponding to a singular vale σ. Strike a row of the matrix $M - \sigma^2 I$. Then the vector of coefficients is the vector of minors of order $n-1$ of the reduced matrix normalized so that their sum of squares is one. Coefficients of the y-substitution may be obtained analogously from $N - \sigma^2 I$. This only works if the singular value σ is simple.

Erhard Schmidt was the first person to use integral equations with unsymmetric kernels to solve the infinite dimensional analogue of singular value decomposition. But he went beyond the mere existence of decomposition by showing how it can be used to obtain optimal, low rank approximations to an operator. E. Schmidt's two contributions to singular value decomposition are its generalization to function space and his approximation theorem. Schmidt's approach is the same as Beltrami's; however, because he worked in finite dimensional spaces of functions he could not appeal to previous result on quadratic forms.

Schmidt began with a kernel $A(s, t)$ that is continuous and symmetric on $[a, b] \times$

$[a, b]$. A continuous, nonvanishing function $\phi(s)$ satisfying

$$\phi(s) = \lambda \int_a^b A(s,t)\phi(t)dt$$

is said to be an eigenfunction of A corresponding to the eigenvalue λ. By the assertions of Schmidt, the kernel A has at least one real eigenfunction. Each eigenvalue of A has at most a finite number of linearly independent eigenfunctions. The kernel A also has an orthonormal system of eigenfunctions; that is, a sequence $\phi_1(s), \phi_2(s), \phi_3(s), ...$ of orthonormal eigenfunctions such that every eigenfunction can be expressed as a linear combination of a finite number of the $\phi_j(s)$. The eigenvalues satisfy

$$\int_a^b \int_a^b (A(s,t))^2 dsdt \geq \sum_i \frac{1}{\lambda_i^2}$$

which implies that the sequence of eigenvalues is bounded.

Schmidt now allows $A(s,t)$ to be unsymmetric and calls any nonzero pair $u(s)$ and $v(s)$ satisfying $u(s) = \lambda \int_a^b A(s,t)v(t)dt$ and $v(s) = \lambda \int_a^b A(s,t)u(t)ds$ a pair of adjoint eigenfunctions corresponding to the eigenvalue λ. The symmetric kernels were then introduced; that is $\overline{A}(s,t) = \int_a^b A(s,r)A(t,r)dr$ and $\underline{A}(s,t) = \int_a^b A(r,s)A(r,t)dr$ and shows that if $u_1(s), u_2(s), u_3(s), ...$ is an orthonormal system for $\overline{A}(s,t)$ corresponding to the eigenvalues $\lambda_1^2, \lambda_2^2, \lambda_3^2, ...$ then the sequence defined by $v_i(t) = \lambda_i \int_a^b A(s,t)u(s)ds$, $i = 1, 2, 3, ...$ is an orthonormal system for $\underline{A}(s,t)$.

Schmidt then went on to consider the expansion of functions in series of eigenfunctions and deduced an expression which according to him, “corresponds to the canonical decomposition of a bilinear form.”

Schmidt's Approximation Theorem

The problem Schmidt sets out to solve is to find the best approximation to A if the form $A \cong \sum_{i=1}^{k} x_i y_i^T$ in the sense that $\| A - \sum_{i=1}^{k} x_i y_i^T \| = min$. In other words,he is looking for the best approximation of rank not greater than k.
He started by noting that if

$$A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T \tag{2.8}$$

then $\| A - A_k \|^2 = \| A \|^2 - \sum_{i=1}^{k} \sigma_i^2$. If for arbitrary x_i and y_i

$$\| A - \sum_{i=1}^{k} x_i y_i^T \| \geq \| A \|^2 - \sum_{i=1}^{k} \sigma_i^2, \tag{2.9}$$

the A_k will be the desired approximation.
Without loss of generality it may be assume that the vectors $x_1, x_2, ...x_k$ are orthonormal. We can use Gram-Schmidt orthogonalization to express them as linear combinations of orthonormal vectors even if they are not orthonormal (Bradley, 1975). Substituting these expressions into $\sum_{i=1}^{k} x_i y_i^T$, and collect terms in the new vectors.
Now

$$\begin{aligned}
&\| A - \sum_{i=1}^{k} x_i y_i^T \| \\
&\qquad = trace((A - \sum_{i=1}^{k} x_i y_i^T)^T (A - \sum_{i=1}^{k} x_i y_i^T)) \\
&\qquad = trace(A^T A + \sum_{i=1}^{k} (y_i - Ax_i)(y_i - A^T x_i)^T - \sum_{i=1}^{k} A^T x_i x_i^T A)
\end{aligned}$$

Since $trace((y_i - A^T x_i)(y_i - A^T x_i)^T) \geq 0$ and $trace(Ax_i x_i^T A^T) = \| Ax_i \|^2$, the result will be established if we show that

$$\sum_{i=1}^{k} \| Ax_i \|^2 \leq \sum_{i=1}^{k} \sigma_i^2$$

Let $V = (V_1 V_2)$ where V_1 has k columns, and let $\Sigma = diag(\Sigma_1, \Sigma_2)$ be a conformal partition of Σ. Then

$$\| Ax_i \|^2 = \sigma_k^2 + (\| \Sigma_1 V_1^T x_i \|^2 - \sigma_k^2 \| V_1^T x_i \|^2) - (\sigma_k^2 \| V_1^T x_i \|^2) - \sigma_k^2 (1 - \| V^T x_i \|) \tag{2.10}$$

The last two terms are clearly nonnegative. Hence

$$\begin{aligned}
\sum_{i=1}^{k} \| Ax_i \|^2 &\leq k\sigma_k^2 + \sum_{i=1}^{k} (\| \Sigma_1 V_1^T x_i \|^2 - \sigma_k^2 \| V_1^T x_i \|^2) \\
&= k\sigma_k^2 + \sum_{i=1}^{k} \sum_{j=1}^{k} (\sigma_j^2 - \sigma_k^2) |V_j^T x_i|^2 \\
&= \sum_{j=1}^{k} (\sigma_k^2 + (\sigma_j^2 - \sigma_k^2) \sum_{i=1}^{k} |V_j^T x_i|^2) \\
&\leq \sum_{j=1}^{k} (\sigma_k^2 + (\sigma_j^2 - \sigma_k^2)) \\
&= \sum_{j=1}^{k} \sigma_j^2
\end{aligned}$$

which establishes the result.

The generalization to function spaces and the approximation theorem are the two contributions of Schmidt to singular value decomposition. An important difference in Schmidt's version of the decomposition is the treatment of null-vectors of A. The crowning glory of Schmidt's work is his approximation theorem, which is nontrivial to conjecture and difficult to prove from scratch. An important application of the approximation theorem is the determination of the rank of a matrix in the presence of error. We will examine the more elegant approach of Weyl.

According to (Stewart, 1992), Hermann Weyl also used integral equations to solve the singular value decomposition. His contributions to theory of the singular value decomposition were to develop a general perturbation theory and use it to give an elegant proof of the approximation theorem. Although Weyl treated integral equations with kernels, in a footnote on Schmidt contribution he states "E. Schmidt's theorem by the way treats arbitrary kernels. However our proof can also be applied directly to this more general case" (Weyl, 1912). Weyl did not actually write about the development of unsymmetric kernels.

The heart of Weyl's development is a lemma concerning the singular values of a perturbed matrix. Specifically, if $B_k = XY^T$, where X and Y have k columns $((B_k) \leq k)$, then

$$\sigma_1(A - B_k) \geq \sigma_{k+1}(A)$$

where $\sigma_i(\cdot)$ denotes the i^{th} singular value of its argument.

Since Y has k columns, there is a linear combination

$$v = \gamma_1 v_1 + \gamma_2 v_2 + \ldots + \gamma_{k+1} v_{k+1}$$

of the first $k+1$ columns of V (from the singular value decomposition of A) such that $Y^T v = 0$.

Assuming that $\| v \| = 1$ or equivalently that $\gamma_1^2 + \gamma_2^2 + ... + \gamma_{k+1}^2 = 1$. It follows that

$$\begin{aligned}\sigma_1^2(A-B) &\geq v^T(A-B)^T(A-B)v \\ &= v^T(A^TA)v \\ &= \gamma_1^2\sigma_1^2+\gamma_2^2\sigma_2^2+...+\gamma_{k+1}^2\sigma_{k+1}^2 \\ &\geq \sigma_{k+1}\end{aligned}$$

According to Weyl(1912), the following assertions hold true; if $A = A' + A''$ then

$$\sigma_{i+j-1} \leq \sigma_i' + \sigma_j'' \tag{2.11}$$

where the σ' and σ'' are the singular values of A' and A'' arranged in order of magnitude. Also, if we set $A' = A - B_k$ and $A'' = B_k$, where B_k has rank k. Since $\sigma_{K+1}(B_K) = 0$ we have by substituting $j = k + 1$ in (2.11)

$$\sigma_i(A - B_k) \geq \sigma_{k+1},$$

for $i = 1, 2, 3, ...$
As a corollary to this result we obtain

$$\parallel A - B_k \parallel^2 \geq \sigma_{k+1}^2 + \sigma_{k+2}^2 + ... + \sigma_n^2.$$

This inequality establishes the approximation theorem. With Weyl's contribution, the theory of Singular Value Decomposition can be said to have matured. In 1902, L. Autonne extended the Singular Value Decomposition to complex matrices. He proved that every non singular complex matrix $A \in M_n$ can be written as $A = UP$, where $U \in M_n$ is unitary and $P \in M_n$ is positive definite. In 1913 to

1915 he returned to these ideas and used the similarity of AA^* and A^*A to show that any square complex matrix $A \in M_n$ can be written as $A = U\Sigma V^*$ where $U, V \in M_n$ are unitary and $\Sigma \in M$ is a nonnegative diagonal matrix. He also discovered that if A is nonsingular Hermitian then A can be written as $U\Sigma U^T$ for some unitary U, and a nonnegative diagonal matrix Σ (Autonne, 1913).

Eckart and Young (Eckart & Young, 1936) (Eckart & Young, 1939) extended it to rectangular matrices and rediscovered Schmidt's approximation theorem which is often called Eckart-Young theorem. In 1939 Eckart and Young gave the first complete proof of the singular value decomposition for rectangular complex matrix and they didn't give any name to the numbers σ_k's. The existence proof of the singular value decomposition opens many ways for the mathematician to search for inequalities, properties and applications to this decomposition.

The term "singular value" seems to have come from the literature on integral equations. After the appearance of Schmidt's paper, H. Bateman refers to the numbers that are essentially the reciprocals of the eigenvalues of the kernel as singular values(Bateman, 1908). In 1949, Weyl spoke of the "two kinds of eigenvalues of a linear transformation,"(Weyl, 1949) and in a 1969 translation of a 1965 Russian treatise on non-self adjoint operators Gohberg and Krein (Gohberg & Krein, 1969) refer to the "s-numbers" of an operator. During 1949-50, a remarkable series of papers in the Proceeding of the National Academy of Science established all of the basic inequalities involving singular values and eigenvalues. One of these papers is "Inequalities Between the Two Kinds of Eigenvalues of a Linear Transformation", established by Weyl. In 1950 Poyla gave an alternative proof of a key lemma in Weyl's 1949 paper(also, established by National Academy of Science). In 1954, A. Horn proved that Weyl's 1949 inequalities were sufficient for the existence of a matrix with prescribed singular values and eigenvalues, and in this paper he used the expression "singular values" in the context of matrices.

The singular value decomposition was introduced into numerical analysis by Golub and Kahan who proposed a computational algorithm. The QR algorithm for the singular values of bidiagonal matrices was first derived by Golub in 1968 without reference to the QR algorithm, which has been the workhorse for two decade. It was Golub who formulated this algorithm which has been used for the past two decades.

Recently in 1990, Demmel and Kahan proposed an interesting alternative for Golub's 1968's algorithm. In the last 30 years, the singular value decomposition has become a popular numerical tool in statistical data analysis, signal processing, system identification and control system analysis and design.

2.4 The Singular Value Decomposition

The singular value decomposition is closely associated with the eigenvalue-eigenvector factorization of a symmetric matrix A into the form;

$$A = U\Lambda V^T = (orthogonal)(diagonal)(orthogonal).$$

Here the eigenvalues are in the diagonal matrix Λ, and the eigenvector matrix U and V^T are orthogonal. The diagonal (but rectangular) matrix in the middle can be made nonnegative, denoted by S and its positive entries called the singular values will be $\sigma_1, \sigma_2, ..., \sigma_r$. They fill the first r places on the main diagonal of S and r is the rank of A. The key to working with rectangular matrices is to consider AA^T and A^TA. Singular Value Decomposition can be considered as a generalization of the spectral decomposition of square matrices, to analyze rectangular matrices. Singular Value Decomposition decomposes a rectangular matrix into three simple matrices. Two orthogonal matrices and one diagonal matrix (Strang, 1980). In general, Singular Value Decomposition theorem can be stated as follows: any $m \times n$ matrix A, with $m \geq n$ can be factored into three matrices: U (column orthogonal, $m \times n$ matrix), Λ (diagonal,$n \times n$ matrix) and V (orthogonal $n \times n$ matrix). When A is real, $A = U\Lambda V^T$. For complex matrices, Λ remains real but U and V become unitary. The diagonal elements of Λ matrix are known as the singular values of A. This decomposition is a technique that works well with matrices that are either singular or else numerically very close to singular. Singular Value Decomposition is also used to calculate pseudo-inverses when the natural inverse of the matrix does not exist (Benyah, 2012). Singular Value Decoposition and pseudo-inverses are generally used in statistics for solving least square problems. Data compression using singular value decomposition is one of the standard applications in image processing.

2.4.1 Geometric Interpretation of the SVD

The purpose of this section of the study is to give an overview of the geometric interpretation of the singular value decomposition of real matrix.

Theorem 2.1 *The image of a unit circle C in R^2 under any nonsingular matrix $A \in R^{2\times 2}$ is an ellipse in R^2.*

Proof. Let v be the unit vector in R^2. That is $v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$ and let A be a nonsingular 2×2 matrix $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$.

We will show that Av is an ellipse.

To prove this we let $Av = u$. Then $v = A^{-1}u$, where $u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$

Since v is a unit vector $v = A^{-1}u$ gives

$$\begin{aligned} 1 = \parallel v \parallel_2^2 &= \parallel A^{-1}u \parallel_2^2 \\ &= (A^{-1}u)^T(A^{-1}u) \qquad (2.12) \\ &= u^T(A^{-1})^T A^{-1}u. \end{aligned}$$

Substituting

$$A^{-1} = \frac{1}{ad - bc}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

into (2.12) we obtain

$$(d^2 + c^2)u_1^2 + (b^2 + a^2)u_2^2 - 2(db + bc)u_1u_2 - (ad - bc)^2 = 0 \qquad (2.13)$$

Equation (2.13) is of the form

$$AX^2 + BXY + CY^2 + DX + EY + F = 0, \qquad (2.14)$$

which is the general equation of a conic whose axes are rotated so that they are not parallel to the x and y-axis.

If the discriminant $B^2 - 4AC < 0$, then equation(2.14) is an ellipse.

From equation(2.13) and (2.14)

$$\begin{aligned} B^2 - 4AC &= (-2(db + ac))^2 - 4(d^2 + c^2)(b^2 + a^2) \\ &= -2(cd - 2bd)^2 < 0 \end{aligned}$$

Therefore Av is an ellipse.

Theorem 2.2 *(Blank, Krikorian, & Spring, 1989) Let* S^{n-1} *be the unit sphere in* $R^n : S^{n-1} = \{x \in R^n :\| x \|_2 = 1\}$. *Let* AS^{n-1} *be the image of* S^{n-1} *under* $A : AS^{n-1} = \{Ax : x \in R^n and \| x \|_2 = 1\}$. *Then* AS^{n-1} *is an ellipsoid centered at the origin of* R^n *with principal axes* $\sigma_i u_i$

Proof. Let us assume that A is square and nonsingular. Since V is orthogonal, it maps unit vector to other vectors, that is, $V^T S^{n-1} = S^{n-1}$.
Since $v \in S^{n-1}$ if and only if $\| v \|_2 = 1$, $w \in \Sigma S^{n-1}$ if and only if $\| \Sigma^{-1} w \|_2 = 1$
Hence,

$$\sum_{i=1}^{n} (\frac{w_i}{\sigma_i})^2 = 1$$

which is an ellipsoid with principal axes $\sigma_i e_i$ where e_i is the i^{th} column of the identity matrix. Finally multiplying each $w = \Sigma v$ by U just rotates the ellipse so that each e_i becomes u_i the i^{th} column of U. This completes the proof when the matrix A is square and nonsingular.
Now let $A : R^m \longrightarrow R^n$. First we restrict A to orthogonal complement of its null space in R^m. It is known (Strang, 1980) that matrix A is nonsingular on this subspace. Using the above procedure we can find orthogonal bases $v_1, v_2, ..., v_k$ of this subspace in R^m and $u_1, u_2, ..., u_k$ of the range space of A on R^n. Then extend these sets to orthonormal bases of R^m and R^n, respectively.

2.4.2 Low Rank Approximations of a Matrix Using SVD

The overview of the known results on best approximation of a matrix by low rank matrices is the purpose of this section of the study. The Frobenius matrix norm and the 2-norm will be used here.

Theorem 2.3 *(Bau & Trefthen, 1997) For any* k *with* $0 \leq k \leq r$, *where* $r =$

rank(A) and define

$$A = \sum_{i=1}^{r} \sigma_i u_i v_i^T$$

and

$$A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T;$$

if $k = min\{m, n\}$, *define* $\sigma_{k+1} = 0$. *Then*

$$\| A - A_k \|_2 = \inf_{B \in R^{m \times n}} \| A - B \|_2 = \sigma_{k+1}.$$

where $rank(B) \leq k$

Proof. Given that

$$A_k = \sum_{i=1}^{k} \sigma_i u_i v^T,$$

we can also write

$$A_k = U S_k V^T,$$

where $S_K = diag(\sigma_1, \sigma_2, \sigma_3, ..., \sigma_k, 0, ..., 0)$ and U, V are orthogonal matrices. We have

$$\begin{aligned} \| A - A_k \|_2 &= \| \sum_{i=k+1}^{r} \sigma_i u_i v_i^T \|_2 \\ &= \| U(diag(0, ..., \sigma_{k+1}, ..., \sigma_r))V^T \|_2 \\ &= \sigma_{k+1} \end{aligned}$$

To show that there is no matrix of rank k closer to A, we assume B to be any rank k matrix so its null space has dimension $r - k$. The space spanned by $\{v_1, v_2, ..., v_{k+1}\}$ has dimension $k + 1$. Since the sum of their dimensions is $(r - k) + (k + 1) > r$ these two spaces must overlap. Let h be a unit vector in their intersection. Then

$$
\begin{aligned}
\| A - A_k \|_2^2 \geq \| (A-B)h \|_2^2 &= \| Ah \|_2^2 \\
&= \| USV^T h \|_2^2 \\
&= \| S(V^T h) \|_2^2 \\
&= \sigma_{k+1}^2 \| V^T h \|_2^2 = \sigma_{k+1}^2 .
\end{aligned}
$$

A similar result holds true when the Frobinius matrix norm is used.

Theorem 2.4 *(Bau & Trefthen, 1997). For any k with $0 \leq k \leq r$, the matrices A_k and A of theorem (2.3) also satisfy*

$$\| A - A_k \|_F = \inf_{B \in R^{m \times n}} \| A - B \|_F = \sqrt{\sum_{i=k+1}^{r} \sigma_i^2}.$$

where $rank(B) \leq k$

Proof. Let U and V be orthogonal matrices. Then $\| B \|_F = \| U^T BV \|_F$. That is

$$
\begin{aligned}
\| A - B \|_F &= \| U^T (A-B)V \|_F \\
&= \| U^T AV - U^T BV \|_F \\
&= \| S - S_k \|_F ,
\end{aligned}
$$

where $A = USV^T$ and $S_k = U^T BV$. That is

$$\| A - B \|_F = \sqrt{\sum_{i=1}^{r} |\sigma_i - \sigma_j|^2}.$$

for $1 \leq j \leq k$

It follows now that

$$\begin{aligned} \| A - A_k \|_F &= \min \| A - B \|_F \\ &= \min \| S - S_k \|_F . \\ &= \sqrt{\sum_{i=k+1}^{r} \sigma_i^2}. \end{aligned}$$

The low rank approximation is normally used in image processing, specifically image compression. The rank of matrix A is equal to the number of its nonzero singular values(Leon, 1996). In many applications, the singular values of a matrix decrease quickly with increasing rank. This propriety allows us to reduce the noise or compress the matrix data by eliminating the small singular values or the higher ranks. When an image is SVD transformed, it is not compressed, but the data take a form in which the first singular value has a great amount of the image information. With this, we can use a few singular values to represent the image with little differences from the original.

2.5 Image Processing

Image processing has received enormous focus in the last century. This is mainly related to the wide area of its applications. The other important feature of image processing tasks is their triviality when we compare them with what our visual system performs. The first image processing research projects were devoted to processing binary and then grayscale images. This was the clear choice at the time, because time had to be spent on developing the first color recording camera. Though, decades passed before digital cameras of acceptable quality and resolution came into market. Working on color image processing was presumed to be unnecessary even in the last decade.

When researchers started to think about color image processing, the first algorithms were just parallelized versions of the available grayscale algorithms. At that time, it was just common to think of a RGB color image as a set of three parallel grayscale images. Unfortunately, this missunderstanding is extensively common in todays image processing community. When working with color images, one has to face the correlation between color components and the distribution of energy among them. This twofold phenomena is both a threat and an opportunity. Thanks to the physical evidences, principal component analysis has proved to be an appropriate tool for both decorrelating color components and compacting the energy. As such, it is the time to quit working with fixed color transformations and face the data-adaptive transformation of principal component analysis(PCA) which seek to find k "principal axes" which are orthonormal coordinate system that can capture most of the variance in data.

Image Processing (face verification) plays an important role in biometrics based personal identification. A biometrics verification system is designed to verify or recognize the identity of a living person on the basis of his/her physiological characters, such as face, fingerprint, and iris, or some aspects of behavior such as handwriting or keystroke pattern. According to Steve Lawrence, the biometrics verification technique acts as an efficient method and has wide applications in the areas of information retrieval, automatic banking, control of access to security areas, and buildings. Compared with other biometric verification techniques, face recognition has the advantages of being passive and non-intrusive(Lawrence, Giles, Tsoi, & Back, 1998).

Great progress has been made in face recognition in the past years. For almost all previously proposed techniques, the success of face recognition depends on the solution of two problems of representation and matching (Zhang, Yan, & Lades, 1997). The representation of a pattern can be considered as feature extraction in pattern recognition. In Hong(1991), image features are divided into four groups: visual features, statistical pixel features, transform coefficient features, and algebraic features. The algebraic features represent intrinsic properties of an image and have good stability. Hong suggested that the algebraic features are valid features in object recognition such as face recognition and proposed a singular value decomposition (SVD) based recognition method which uses the singular values as the feature vectors (Hong, 1991). The effectiveness of the Singular Value Decomposition has been tested in (Hong, 1991) and (Yong-Qing, 1991) respectively. In Hong(1991), an error rate of 42.47% was recorded which was thought to be caused by the statistical limitations of the small samples. Cheng (Yong-Qing, 1991) proposed a human face recognition method based on the statistical model of small sample size that also used the singular values as the face features.

In his paper, Hong constructed an optimal discriminate transformation to transform an original space of singular value (SV) vectors into a new space whose dimension is significantly lower than that of the original space to minimize the small sample size effect. That approach was tested on 64 facial images of eight people. Good discrimination ability was obtained with an accuracy rate of 100% (Yong-Qing, 1991). It should be noted that in order to make the method independent of translation, rotation and scaling, the images were represented by Goshtasby's shape matrices. The Goshtasby's shape matrices are invariant to translation, rotation, and scaling of the facial images and are obtained by polar quantization of the shape (Cheng, Liu, Yang, & Wang, 1992). The above two methods have never been tested with large face databases and their effectiveness with large databases remains unknown. Especially when there are variations in lighting and viewpoint. Both methods use only singular values as face features.

Face recognition is one of the most important biometrics which seems to be a good compromise between actuality and social reception and balances security and privacy well. It has a variety of potential applications in information security law enforcement and access controls. Face recognition systems fall into two categories: verification and identification. Face verification is 1:1 match that compares a face images against a template face image. On the other hand face identification is 1: N problem that compares a probe face image against all image templates in a face database (Hasan, Jouhar, & Alwan, 2012). Face recognition is a very difficult problem due to a substantial variations in light direction (illumination) , different face poses , diversified facial expressions , Aging (changing the face over time) and Occlusions (like glasses, hair, cosmetics). So the building of an automated system that accomplishes such objectives is very challenging. In last decades many systems with recognition rate greater than 90% has been done however a perfect system with 100% recognition rate remains a challenge. Face recognition algorithms are divided according to (Kachare & Inamdar, 2010) and

(Patil, Kolhe, & P.M, 2010) into three categories. The Holistic, Feature based and the Hybrid methods. The Holistic methods identify a face using the whole face images as input and extract the overall features of the face. Feature based methods uses the local facial features like eyes, mouths, and fiducial points for recognition. The Hybrid methods uses both feature based and holistic features to recognize a face. These methods have the potential to offer better performance than individuals.

2.5.1 Image File Formats

In the context of Digital Image processing, an image file format is a standard way to organize and store image data. It defines how the data is arranged and the type of compression that is used. There are several formats file in which image files can be compressed (V.J & Dasgupta, n.d.). These include Bitmap(BMP), Graphics Interchange Format(GIF), Portable Network Graphics(PNG), Tagged Image File Format(TIFF), Portable Pix Map(PPM), Portable Grayscale Map(PGM) and Joint Photographic Expert Group(JPEG):

Bitmap (BMP):

Windows Bitmap or BMP files are image files within the Microsoft Windows Operating System. Bitmap files are not very popular as they do not scale or compress the images well. Being oversized, this format is not web friendly.

Graphics Interchange Format(GIF):

Graphics Interchange Format is a popular image format on the internet. This is because its file size is relatively small compared to other image compression types. Graphics Interchange Format is most suitable for graphics, animations, diagrams and cartoons. Graphics Interchange Format was introduced in the 1980's to allow high-quality, high-resolution graphics to be displayed on a variety of graphics hardware and was intended as an exchange and display mechanism for graphics images. The rise of the Internet and in particular the web saw the Graphics Interchange Format usage explode. A Graphics Interchange Format graphic is stored as a sequence of pixels with 256 color values from a image specific color palette. Dithering reduces the visual impact of the reduction in number of colors.

Portable Network Graphics(PNG):

This format is designed specifically for web applications. This format is lossless so it does not lose quality and detail after image compression. PNG format is not suitable for large images because they tend to generate a very large file. The Portable Network Graphics (PNG) format was designed to replace the older and simpler Graphics Interchange Format and, to some extent, the much more complex Tagged Image File Format(TIFF) format. For image editing PNG provides a useful format for the storage of intermediate stages of editing. PNG's compression is fully lossless and since it supports up to 48-bit truecolor(RGB) or 16-bit grayscale saving, restoring and re-saving an image will not degrade its quality. Unlike TIFF, the PNG specification leaves no room for implementors to pick and choose what features they'll support; the result is that a PNG image saved in one application is readable in any other PNG-supporting application. For the Web, PNG really has three main advantages over GIF: variable transparency, cross-platform control of image brightness, and a method of progressive display. PNG also compresses better than GIF in almost every case.

Tagged Image File Format (TIFF):

It is recommended especially for text, black and white images. Tagged Image File Format is very flexible; it can be lossy or lossless. It is a rich format and is supported by many imaging programs. It is the standard format for printing, scanned documents and optical character recognition since it does not have any artifacts. Drawbacks of this format include long transfer time, huge disc space consumption and slow loading time.

Portable Pix Map (PPM):

It is a very old image format that can represent any ordinary colour image. Portable Pix Map files are basically plain text files making it one of the simplest formats. The PPM format is not intended to be an archival format, so it does not need to be too storage efficient.

Portable Grayscale Map (PGM):

Portable Grayscale Map format represents a grayscale graphic image. The PGM format is a lowest common denominator grayscale file format. It is designed to be extremely easy to learn and write programs for. (It's so simple that most people will simply reverse engineer it because it's easier than reading this specification). There are many pseudo-PGM formats in use where everything is as specified herein except for the meaning of individual pixel values. For most purposes, a PGM image can just be thought of an array of arbitrary integers, and all the programs in the world that think they are processing a grayscale image can easily be tricked into processing something else.

Joint Photographic Expert Group (JPEG):

JPEG (pronounced "jay-peg") is a standardized image compression mechanism. Joint Photographic Experts Group (JPEG), is the original name of the committee that wrote the standard. Joint Photographic Expert Group file format differs from other file formats as it is lossy. JPEGs compression technology reduces the true quality of the image in order to achieve its striking file size reduction. JPEG compression is lossy, meaning that the decompressed image isn't quite the same as the one you started with. This file format was designed specifically for use with highly detailed or photo realistic images, and is typically applied to rendered images and digitized photographs. It is not suitable for use with rough drafts, line drawings, screen captures and other image types which use sharply defined lines and coloured images. JPEG is designed to exploit known limitations of the human eye, notably the fact that small color changes are perceived less accurately than small changes in brightness. That is, JPEG is intended for compressing images that will be looked at by humans. If you plan to machine-analyze your images, the small errors introduced by JPEG may be a problem for you, even if they are invisible to the eye.A useful property of JPEG is that the degree of lossiness can be varied by adjusting compression parameters. This means that you can trade off file size against output image quality. The quality scale is purely arbitrary; its not a percentage of anything.

Chapter 3

Methodology

3.1 Introduction

This chapter describes the theory of the concept used, derivation and methods of analyzing the available data to satisfy the objectives of the study. It focused on the detail and comprehensive understanding of the Singular Value Decomposition methodology for image processing. Among the aspects that will come under discussion include the methodologies used in the image processing. The algorithms for generating processed images will be written in language of Matlab.

3.2 The Process of Singular Value Decomposition

Singular Value Decomposition is an effective numerical analysis tool used to analyze matrices. Singular Value Decomposition is an optimal matrix decomposition technique in a least square sense that it packs maximum signal energy into as few coefficients as possible. It has the ability to adapt to the variations in local statistics of an image. In Singular Value Decomposition transformation, a matrix can be decomposed into three other matrices that are of the same size as the original matrix. From the view point of linear algebra, an image is an array of non-negative scalar entries that can be regarded as a matrix. Without loss of generality, if A is a square image(matrix), denoted as $A \in R^{n\times n}$, where R represents the real number domain, then the singular value decomposition of A is defined as

$$A = USV^T \tag{3.1}$$

where $U \in R^{n\times n}$ and $V \in R^{n\times n}$ are orthogonal matrices, and $S \in R^{n\times n}$ is a diagonal matrix, as

$$\begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{bmatrix}$$

Here the diagonal elements, $\sigma's$, are referred to as the singular values and satisfy the condition

$$\sigma_1 \leq \sigma_2 \leq \sigma_3 \leq ...\sigma_r \leq \sigma_{r+1} \leq= ... = \sigma_n = 0$$

Generally, in every real $m \times n$ matrix, A, there exist orthogonal matrices U and V such that

$$U^T_{m\times m}A_{m\times n}V_{n\times n} = S_{m\times n} = \begin{bmatrix} S_1 & 0 \\ 0 & 0 \end{bmatrix} \tag{3.2}$$

where S_1 is a nonsingular diagonal matrix. The diagonal entries of S are non-negative and can be arranged in a nonincreasing order. The number of a nonzero diagonal entries of S equals the rank of A.

To justify the assertion above, consider the symmetric positive semidefinite matrix A^TA; its eigenvalues are nonnegative. Represent the eigenvalues of A^TA by $\lambda_1 = \sigma_1^2, \lambda_2 = \sigma_2^2, ..., \lambda_n = \sigma_n$. Denote the set of orthonormal eigenvectors of A^TA corresponding to eigenvalues by $v_1, v_2, ..., v_n$; that is v_1 through v_n are orthonormal and satisfy $A^TAv_i = \sigma_1^2 v_i$, $i = 1, 2, ..., n$ Then

$$v_i^T A^T A v_i = \sigma_i^2 \neq 0, \tag{3.3}$$

for $i = 1, 2, ..., r$ and

$$v_i^T A^T A v_j = \sigma_i^2 = 0, \tag{3.4}$$

for $i = 1, 2, ..., r; j \neq i$

Write $V_1 = (v_1, v_2, ..., v_r)$ and $V_2 = (v_{r+1}, v_{r+2}, ..., v_n)$ where v_1 through v_r are the eigenvectors associated with the nonzero eigenvalues λ_1 through λ_r and $v_{r+1}, ..., v_n$ correspond to the zero eigenvalues. Then

$$\begin{aligned} V_2^T A^T A V_2 &= V_2^T A^T A (v_{r+1}, v_{r+2}, ..., v_n) \\ &= V_2^T (0, 0, ..., 0) = 0 \end{aligned}$$

This implies that $AV_2 = 0$, or

$$Av_k = 0, k = r+1, r+2, ..., n \tag{3.5}$$

Next define a set of nonzero vectors $\{u_i\}$ by

$$u_i = \frac{1}{\sigma_i} A v_i \tag{3.6}$$

for $i = 1, 2, ..., r$.

This form an orthonormal set because

$$\begin{aligned} U_i^T U_j &= \frac{1}{\sigma_i}(Av_i)^T \frac{1}{\sigma_j}(Av_j) \\ &= \frac{1}{\sigma_i \sigma_j}(v_i^T A^T A v_j) \\ &= \begin{cases} 0 & \text{when } i \neq j; \\ 1 & \text{when } i = j. \end{cases} \end{aligned} \tag{3.7}$$

Define $U_1 = (u_1, u_2, ..., u_r)$, and choose $U_2 = (u_{r+1}, ..., u_m)$ such that $U = (U_1, U_2)$ is orthogonal. Then for any $k > r$,we have

$$u_k^T A v_i = \sigma_i u_k^T u_i = 0, i = 1, ..., r$$

from equation (3.6). that is by the orthogonality of the vectors of U and $u_k^T v_i = 0$, $i = r + 1, ..., n$ from equation (3.5).

Let $V = (V_1, V_2)$ then

$$U^T AV = \begin{bmatrix} u_1^T \\ u_2^T \\ \vdots \\ u_m^T \end{bmatrix} A(v_1, v_2, ..., v_n)$$

$$= \begin{bmatrix} \frac{1}{\sigma_1} v_1^T A^T \\ \frac{1}{\sigma_2} v_2^T A^T \\ \vdots \\ \frac{1}{\sigma_r} v_r^T A^T \\ u_{r+1}^T \\ \vdots \\ u_m^T \end{bmatrix} A(v_1, v_2, ..., v_n) \tag{3.8}$$

$$U^T AV = \begin{bmatrix} a_1 & & & 0 \\ & a_2 & & \\ & & \ddots & \\ & & & a_r \\ 0 & & & 0 \end{bmatrix} = S = \begin{bmatrix} S_1 & 0 \\ 0 & 0 \end{bmatrix}$$

where $a_i = \frac{1}{\sigma_i}\sigma_i^2$ for $i = 1, 2, ..., r$ and S_1 is the diagonal matrix whose diagonal entries are the nonzero singular values of A.

The statement about the rank of A becomes obvious, because

$$rank(A) = rank(USV^T) = rank(S) = r.$$

The decomposition $A = USV^T$ is known as the singular value decomposition or SVD of A. The column vectors of U and V are called respectively, the left singular vectors and right singular vectors of A.

There are $k = min(m, n)$ singular values of A. If r is the rank of A, then there are r positive singular values of A. These are the positive square roots of the nonzero eigenvalues of A^TA (or AA^T). The remaining $(k-r)$, if $r < k$, singular values are zero. That is, the singular values are unique. However,the singular vectors are not unique. For example, if A has a multiple singular value $\sigma > 0$, then the corresponding columns of the matrix V can be chosen as any orthonormal basis of the space spanned by the eigenvectors associated with the multiple eigenvalue $\lambda = \sigma^2$ of A^TA.

Theorem 3.1 *Let A be an $m \times n$ matrix, $(m \geq n)$. Then*

1. The matrices $(A^TA)_{n\times n}$ and $(AA^T)_{m\times m}$ are symmetric with real and nonnegative eigenvalues.

2. If λ is a nonzero eigenvalue of A^TA corresponding to the eigenvector $\boldsymbol{x}$, then λ is also an eigenvalue of AA^T with corresponding eigenvector $\boldsymbol{Ax}$. In other words, A^TA and AA^T have the same nonzero eigenvalues.

Proof 3.1

1. $(A^TA)^T = A^T(A^T)^T = A^TA$, and so A^TA is symmetric. Similarly, for AA^T, $(AA^T)^T = (A^T)^TA^T = AA^T$.

Let $\mathbf{x}$ be an eigenvector of A^TA corresponding to a nonzero eigenvalue λ. Then

$$A^TA\mathbf{x} = \lambda\mathbf{x} \tag{3.9}$$

Multiplying through equation (3.9) on the left by $\mathbf{x}^T$ yeilds

$$\mathbf{x}^T A^T A\mathbf{x} = \lambda \mathbf{x}^T \mathbf{x}$$

$$(A\mathbf{x})^T (A\mathbf{x}) = \lambda \parallel \mathbf{x} \parallel_2^2$$

$$\parallel A\mathbf{x} \parallel_2^2 = \lambda \parallel \mathbf{x} \parallel_2^2$$

$$\geq 0$$

Hence

$$\lambda = \frac{\parallel A\mathbf{x} \parallel_2^2}{\parallel \mathbf{x} \parallel_2^2} \geq 0$$

This holds for AA^T.

2. Multiplying through equation (3.9) on the left by A gives

$$A(A^T A)\mathbf{x} = A(\lambda \mathbf{x})$$

$$AA^T (A\mathbf{x}) = \lambda (A\mathbf{x})$$

That is, $A\mathbf{x}$ is an eigenvector of AA^T corresponding to the eigenvalue λ, where $\mathbf{x}$ is an eigenvector of $A^T A$ corresponding to the eigenvalue λ. Hence $A^T A$ and AA^T have the same nonzero eigenvalue λ.

Properties and Observations of the SVD

There are many properties and attributes of Singular Value Decomposition, parts of the properties that are used in this research are presented as follows.

1. The singular values $\sigma_1, \sigma_2, ..., \sigma_n$ of A are unique; however the matrices U and V are not unique.

2. Since $A^T A$ is symmetric, we can find an orthonomal set of eigenvectors v_i, $i = 1, 2, .., n$ such that the orthogonal matrix $V = [v_1, v_2 ... v_n]$ diagonalizes $A^T A$.

3. Similarly, we can find an orthonormal set of eigenvectors u_i, $i = 1, 2, .., m$ such that the orthogonal matrix $U = [u_1, u_2 ... u_m]$ diagonalize $A^T A$. Moreover, $AA^T = USS^T U^T$

4. Comparing the j^{th} column of each side of the equation $AV = US$ we get $Av_j = \sigma_j u_j$, $j = 1, 2, ..., n$. Similarly, $A^T U = VS^T$, and so $A^T u_j = \sigma_j v_j$ for $j = 1, 2, ..., n$ and $A^T u_j = 0$ for $j = 1, 2, ..., m$

5. If the matrix A has rank r, then
   - $v_1, v_2, ..., v_r$ form an orthonormal basis for $R(A^T)$
   - $v_{r+1}, v_{r+2}, ..., v_n$ form an orthonormal basis for $N(A)$
   - $u_1, u_2, ..., u_r$ form an orthonormal basis for $R(A)$
   - $u_{r+1}, u_{r+2}, ..., u_m$ form an orthonormal basis for $N(A^T)$

6. The rank of the matrix A is equal to the number of its nonzero singular values (where singular values are counted according to multiplicity). A similar assumption about eigenvalues is not true. For example, the matrix

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

   has rank 4 even though all of its eigenvalues are 0.

7. In the case that A has rank $r < n$, if we set $U_1 = (u_1, u_2, ..., u_r)$, $V_1 = (v_1, v_2, ..., v_r)$ and define S_1 as the diagonal matrix whose diagonal entries are the nonzero singular values of A, then we obtain the following factorization

called the compact form of the singular value decomposition of A

$$A = U_1 S_1 V_1^T$$

Obtaining the rank of a matrix is useful in many applications of Linear Algebra. One example can be computing the number of solutions of a system of linear equations. In many applications, it is necessary to either determine the rank of a matrix or to determine whether the matrix is deficient in rank. Gaussian elimination is one approach to obtain the rank by reducing the matrix to the echelon form and then counting the number of nonzero rows. However, this approach will often produce errors during the elimination process. The singular value decomposition presents a method for determining how close the given matrix is to a matrix of smaller rank.

3.2.1 The Moore-Penrose Pseudoinverse of a Matrix

If A is an $n \times n$ matrix with linearly independent columns, then A is invertible, and the solution of the linear system $Ax = b$ is $x = A^{-1}b$. On the other hand, if A is an $m \times n$ matrix $(m > n)$ with linearly independent columns, then the system $Ax = b$, $A \in R^{m,n}$ has a unique least-squares solution given by $\hat{x} = (A^T A)^{-1} A^T b$ The pseudo-inverse (also called the Moore-Penrose inverse and denoted by $(A^\dagger)$) of an $n \times m$ matrix $(n > m)$ A is defined by

$$A^\dagger = (A^T A)^{-1} A^T.$$

Singular Value Decomposition provides an intuitive framework to compute a pseudo-inverse $A^\dagger$ of a matrix operator A (Shilov, 1977). If the rank $A = r < n$, then

$$A = USV^T = \begin{bmatrix} U_r & U_{m-r} \end{bmatrix} \begin{bmatrix} D_r & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_r^T \\ V_{n-r}^T \end{bmatrix} \tag{3.10}$$
$$= U_r D_r V_r^T$$

where $U_r = [u_1 u_2 ... u_r]$, $U_{m-r} = [u_{r+1} u_{r+2} ... u_m]$, D_r is the $r \times r$ diagonal matrix with diagonal entries $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > 0$, $V_r = [v_1 v_2 ... v_r]$, $V_{n-r} = [v_{r+1} v_{r+2} ... v_n]$. This factorization of A is called reduced singular value decomposition of A.

Since the diagonal entries of D_r in equation(3.10) are nonzero, the matrix D_r is invertible. In this case, we can use Singular Value Decomposition to obtain a pseudo-inverse operator $A^\dagger$ which exactly inverses the operations of the original matrix A which will satisfy the equation $\bar{x} = A^\dagger b$. The pseudo-inverse $(A^\dagger)$, is given by

$$A^\dagger = VD^{-1}U^T$$
$$= \begin{bmatrix} v_1 & v_2 & ... & v_{n-1} & v_n \end{bmatrix} \begin{bmatrix} \frac{1}{\sigma_1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{\sigma_2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{...} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{\sigma_{n-1}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\sigma_n} & 0 & 0 \end{bmatrix} \begin{bmatrix} u_1^T \\ u_2^T \\ ... \\ u_{m-1}^T \\ u_m^T \end{bmatrix}$$

The activity of the above pseudo-inverse on the target value b can be interpreted as the target vector b is projected onto the basis in the output space. Then, the projected values are reverse-stretched (scaled) with respect to the corresponding inverses of the singular values. We note that some of the basis disappear due to the zero singular values. This disappearance guarantees that the target value

b is now in the column space of the matrix A (Strang, 1980). The result is re-expressed with respect to the basis in the input vector space V . Hence, the Singular Value Decomposition of A already guarantees that we can obtain a pseudo-inverse readily from the factorization which is always possible. When the matrix is not full rank, the pseudo-inverses are used (Golub & Kahan, 1965).

Properties of pseudo-inverse

The Pseudo-inverse of a matrix A, has the following properties.

- $AA^{\dagger}A = A$
- $A^{\dagger}AA^{\dagger} = A^{\dagger}$
- $(AA^{\dagger})^T = AA^{\dagger}$
- $(A^{\dagger}A)^T = A^{\dagger}A$
- For each $x \in \Re^n$, $A^{\dagger}Ax$ is the orthogonal projection of x onto $RowA$.
- For each $b \in \Re^m$, $AA^{\dagger}b$ is the orthogonal projection of b onto $ColA$

3.3 The Singular Value Decomposition and Linear Systems

A set of linear algebraic equations can be written as

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$

where $\mathbf{A}$ is a matrix of coefficients $(m \times n)$ and $\mathbf{b}(m \times 1)$ is some form of a system output vector. The vector $\mathbf{x}$ is what we usually solve for. If $m = n$ then there are as many equations as unknowns, and there is a good chance of solving for $\mathbf{x}$. That is

$$\mathbf{A}^{-1}\mathbf{A}\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$

Here, we simply compute the inverse of $\mathbf{A}$. This can prove to be a challenging task, however, for there are many situations where the inverse of $\mathbf{A}$ does not exist. In these cases we will approximate the inverse via the Singular Value Decomposition which can turn a singular problem into a non-singular one.

3.3.1 Linear Systems with equal number of equations and unknowns

This is the case when matrix $\mathbf{A}$ is square. We have already presented the case when $\mathbf{A}$ is both square and symmetric. But what if it is only square, or more importantly, square and singular or degenerate (i.e., one of the rows or columns of the original matrix is a linear combination of another one). Here again we use

Singular Value Decomposition. Take for example the following matrix

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 2 & 2 \end{bmatrix}$$

This matrix is square but not symmetric. Furthermore it is singular since the determinant $|\mathbf{A}| = 0$. This would imply $\mathbf{A}^{-1}$ does not exist. Using the Singular Value Decomposition, however, we can approximate an inverse. The Singular Value Decomposition approach tells us to compute eigenvalues and eigenvectors from the inner and outer product matrices:

$$\mathbf{A}^T\mathbf{A} = \begin{bmatrix} 5 & 5 \\ 5 & 5 \end{bmatrix}$$

and

$$\mathbf{A}\mathbf{A}^T = \begin{bmatrix} 2 & 4 \\ 4 & 8 \end{bmatrix}$$

The inner and outer product matrices are both symmetric. The eigenvalues from these matrices are $\lambda_1 = 0$ and $\lambda_2 = 10$. Consequently, the singular values of $\mathbf{A}$ are $\sigma_1 = 0$ and $\sigma_2 = \sqrt{10}$. Therefore the rank of $\mathbf{A}$ is 1. The singular value decomposition is then expressed as

$$\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T = \begin{bmatrix} \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{5}} \\ \frac{-1}{\sqrt{5}} & \frac{2}{\sqrt{5}} \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & \sqrt{10} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}^T = \begin{bmatrix} 1 & 1 \\ 2 & 2 \end{bmatrix}$$

3.3.2 Underdetermined Systems

This is the case where the matrix $\mathbf{A}$ in the system $\mathbf{Ax} = \mathbf{b}$ is $m \times n$ with $m < n$. That is, the number of equations are less than the number of unknowns. For example, if

$$\mathbf{A} = \begin{bmatrix} 3 & 1 & 1 \\ -1 & 3 & 1 \end{bmatrix},$$

then

$$\mathbf{A}^T\mathbf{A} = \begin{bmatrix} 10 & 0 & 2 \\ 0 & 10 & 4 \\ 2 & 4 & 2 \end{bmatrix}$$

and

$$\mathrm{AA}^T = \begin{bmatrix} 11 & 1 \\ 1 & 11 \end{bmatrix}$$

The eigenvalues from $\mathbf{A}^T\mathbf{A}$ are $\lambda_1 = 12$, $\lambda_2 = 10$ and $\lambda_3 = 0$. The eigenvalues from $\mathbf{A}\mathbf{A}^T$ are $\lambda_1 = 12$ and $\lambda_2 = 10$. Consequently, the non-zero singular values of $\mathbf{A}$ are $\sigma_1 = \sqrt{12}$ and $\sigma_2 = \sqrt{10}$. Therefore the rank of $\mathbf{A}$ is 2. The singular value decomposition is then expressed as

$$\mathbf{A} = \mathbf{USV}^T = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \sqrt{12} & 0 & 0 \\ 0 & \sqrt{10} & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{6}} & \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{30}} \\ \frac{2}{\sqrt{6}} & \frac{-1}{\sqrt{5}} & \frac{1}{\sqrt{30}} \\ \frac{2}{\sqrt{6}} & 0 & \frac{-5}{\sqrt{30}} \end{bmatrix}^T \begin{bmatrix} 3 & 1 & 1 \\ -1 & 3 & 1 \end{bmatrix}$$

3.3.3 Overdetermined Systems

This is the case where the matrix $\mathbf{A}$ in the system $\mathbf{Ax} = \mathbf{b}$ is $m \times n$ with $m > n$. That is, the number of equations are more than the number of unknowns. When we have a set of linear equations with more equations than unknowns, and we wish to solve for the vector $\mathbf{x}$, we usually do so in a least-squares sense. For example, if

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 0 & 0 \end{bmatrix},$$

then $\mathbf{A}^T\mathbf{A} \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}$ and $\mathbf{A}\mathbf{A}^T = \begin{bmatrix} 2 & 2 & 0 \\ 2 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

The eigenvalues from $\mathbf{A}^T\mathbf{A}$ are $\lambda_1 = 4$ and $\lambda_2 = 0$. The eigenvalues from $\mathbf{A}\mathbf{A}^T$ are $\lambda_1 = 4$, $\lambda_2 = 0$ and $\lambda_3 = 0$. Consequently, the non-zero singular values of $\mathbf{A}$ are $\sigma_1 = 4$ and $\sigma_2 = 0$. Therefore the rank of $\mathbf{A}$ is 1. The singular value decomposition is then expressed as

$$\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \sqrt{4} & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix}$$

3.3.4 Overdetermined System: Least-Squares Solution

The Singular Value Decomposition provides the most numerically efficient approach for solving the least-square problem

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{3.11}$$

where $\mathbf{A} \in R^{m \times n}$, $m \geq n$, $\mathbf{x} \in R^n$, $\mathbf{b} \in R^m$.

Assume that the singular values are arranged so that $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \ldots \geq \sigma_n \geq 0$ and that the rank of $\mathbf{A}$ is n. Then a least-squares solution $\mathbf{x}_0$ of equation(3.11) satisfies

$$\mathbf{A}^T\mathbf{A}\mathbf{x}_0 = \mathbf{A}^T\mathbf{b} \tag{3.12}$$

If we write $\mathbf{b} \in R^m$ as

$$\mathbf{b} = \sum_{i=1}^{m} b_i u_i$$

where the u_i's are orthonormal eigenvectors of AA^T , then by the orthonormality of the u_i's, we have

$$b_i = u_i^T \mathbf{b}$$

Similarly, we can expand $\mathbf{x}_0 \in R^n$ as

$$\mathbf{x}_0 = \sum_{i=1}^{n} a_i v_i$$

where the v_i's are orthonormal eigenvectors of $A^T A$. Then,

$$\begin{aligned} \mathbf{A}^T \mathbf{A} \mathbf{x}_0 &= \mathbf{A}^T \mathbf{A} \sum_{i=1}^{n} a_i v_i \\ &= \sum_{i=1}^{n} a_i \sigma_i^2 v_i \end{aligned} \tag{3.13}$$

Also,

$$\begin{aligned} \mathbf{A}^T \mathbf{b} &= \mathbf{A}^T \sum_{i=1}^{m} b_i u_i \\ &= \sum_{i=1}^{n} b_i \sigma_i v_i \end{aligned} \tag{3.14}$$

since $\mathbf{A}^T u_i = 0, i = n+1, n+2, ..., m$. Substituting equation(3.13) and (3.14) into (3.12) gives

$$a_i = \frac{b_i}{\sigma_i} = \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i}$$

and therefore

$$\mathbf{x}_0 = \sum_{i=1}^{n} \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i} \mathbf{v}_i \tag{3.15}$$

as the solution to the linear system.

3.4 SVD Approach for Face Image Recognition

Singular Value Decomposition treats a set of known faces as vectors in a subspace, called "face space", spanned by a small group of "base-faces" (Zeng, 2006). Face image recognition in singular value decomposition is performed by projecting a new image onto the face space, and then classifying the face by comparing its coordinates (position) in face space with the coordinates (positions) of known faces. However, the Singular Value Decomposition approach has better numerical properties than Principal Component Analysis (PCA). In the case of Singular Value Decomposition, we redefined the matrix A as set of the training face. Assume each face image has $m \times n = M$ pixels, and it is represented as an $M \times 1$ column vector f_i , a 'training set' S with N number of face images of known individuals forms an $M \times N$ matrix:

$$S = [f_1, f_2, f_3, ..., f_N] \tag{3.16}$$

.

The mean image $\overline{f}$ of set S, is given by

$$\overline{f} = \frac{1}{N}\sum_{i=1}^{N} f_i \tag{3.17}$$

Subtracting this mean from the original faces gives

$$a_i = f_i - \overline{f}, \tag{3.18}$$

$$i = 1, 2, 3, ..., N.$$

This gives another matrix

$$A = [a_1, a_2, ..., a_N] \tag{3.19}$$

with dimension $M \times N$.

We then apply the concept of Singular Value Decomposition to decompose A into USV^T.

Since $\{u_1, u_2, ..., u_r\}$ form an orthonormal basis for $R(A)$, the range (column) subspace of matrix A and matrix A is formed from a training set S with N face images, $R(A)$ is called a 'face subspace'.

Let $X = [x_1, x_2, ..., x_r]^T$ be the coordinates (position) of any $m \times n$ face image f in the face subspace. Then it is the scalar projection of $f - \overline{f}$ onto the basefaces:

$$X = [u_1, u_2, ..., u_r]^T [f - \overline{f}] \tag{3.20}$$

This coordinate vector X is used to find which of the training faces best describes the face f. That is to find some training face f_i, where $i = 1, 2, ..., N$, that minimizes the distance:

$$\begin{aligned} \varepsilon_i &= \| X - X_i \|_2 \\ &= [(X - X_i)^T (X - X_i)]^{\frac{1}{2}} \end{aligned} \tag{3.21}$$

where X_i is the coordinate vector of f_i, which is the scalar projection of $f_i - \overline{f}$ onto the basefaces:

$$X_i = [u_1, u_2, ..., u_r]^T [f_i - \overline{f}] \tag{3.22}$$

A face f is classified as face f_i when the minimum ε_i is less than some predefined threshold ε_0. Otherwise the face f is classified as "unknown face". If f is not a face, its distance to the face subspace will be greater than 0. Since the vector projection of $f - \overline{f}$ onto the face space is given by

$$f_J = [u_1, u_2, ..., u_r]X \tag{3.23}$$

where X is given in equation (3.20).

The distance of f to the face space is the distance between $f - \overline{f}$ and the projection f_J onto the face space:

$$\begin{aligned} \varepsilon_f &= \| (f - \overline{f}) - f_J \|_2 \\ &= [(f - \overline{f} - f_J)^T (f - \overline{f} - f_J)]^{\frac{1}{2}} \end{aligned} \tag{3.24}$$

If ε_f is greater than some predefined threshold ε_1, then f is not a face image.

Steps to Conduct Face Recognition with SVD

1. Obtain a training set S with N face images of known individuals.

2. Compute the mean face $\overline{f}$ of S by using equation (3.17)

3. Form the matrix A in equation (3.19) with the computed mean face $\overline{f}$. Calculate the singular value decomposition of A as shown in equation (3.1).

4. For each known individual, compute the coordinate vector X_i from equation (3.22). Choose a threshold ε_1 that defines the maximum allowable distance from face space.

5. Determine a threshold ε_0 that defines the maximum allowable distance from any known face in the training set S.

6. For a new input image f to be identified, calculate its coordinate vector X from equation (3.20), the vector projection f_J , the distance ε_f to the face space from equation (3.24). If $\varepsilon_f > \varepsilon_1$, then the input image is not a face.

7. If $\varepsilon_f < \varepsilon_1$, compute the distance ε_i to each known individual. If all $\varepsilon_i > \varepsilon_0$, the input image may be classified as unknown face, and optionally used to begin a new individual face. If $\varepsilon_f < \varepsilon_1$, and some $\varepsilon_i < \varepsilon_0$, classify the input image as the known individual associated with the minimum ε_i (X_i), and this image may optionally added to the original training set (Jain & Gautam, 2012).

The flowchart for face recognition with SVD is showed in the figure below.

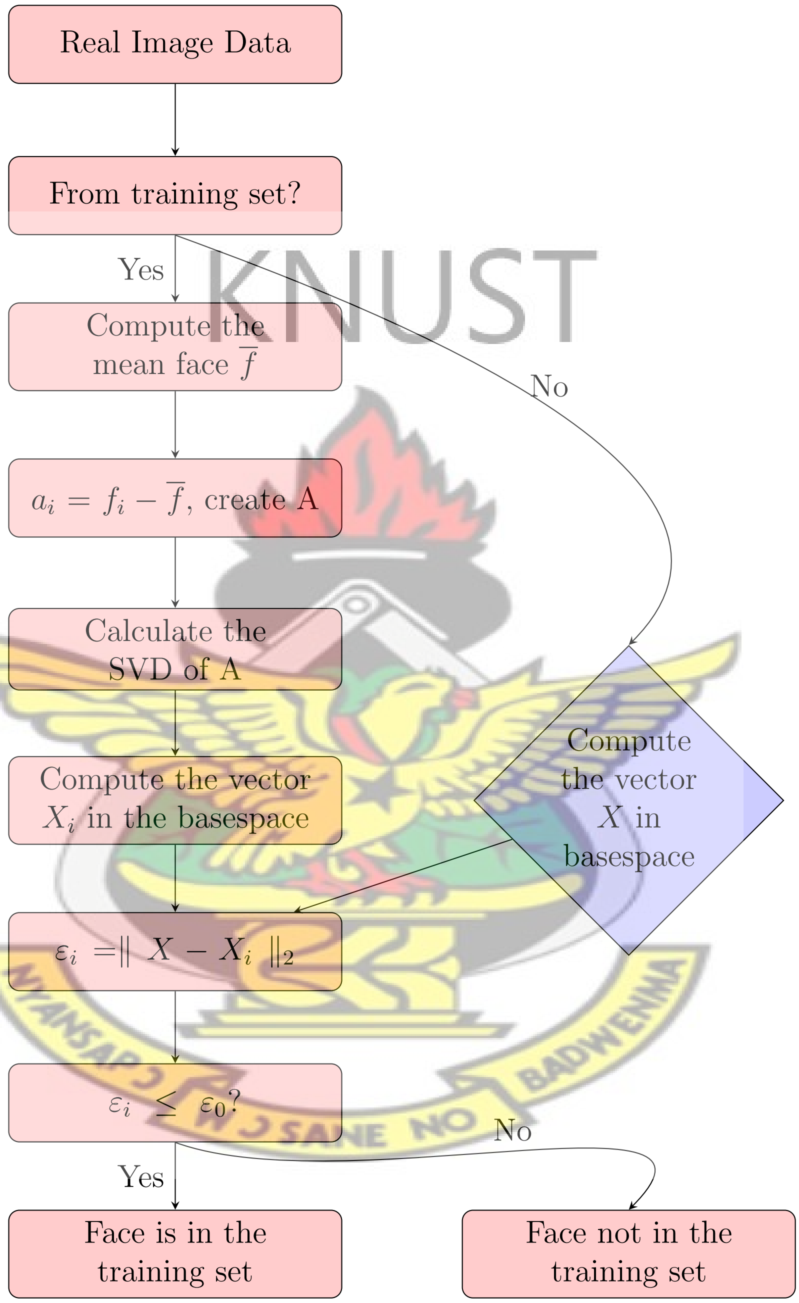

Figure 3.1: Flow chart of Face Recognition with SVD

Chapter 4

Experimentation and Results

4.1 Introduction

The algorithm was implemented using MATLAB R2012a software. MATLAB is a production of MathWorks Co. and can perform algorithm development, data visualization, data analysis, and numeric computation with traditional programming language. Signal processing, image processing, controller design, mathematical computation, etc. may be implemented easily with MATLAB that includes many toolboxes which simplifies generation of algorithm more powerfully. Image Acquisition Toolbox, Image Processing Toolbox and other Toolboxes are used while implementing the algorithm of Face recognition. Image Acquisition Toolbox enables image acquiring from frame grabber or other imaging system that MATLAB supports. This toolbox supports acquiring resolution of frame grabber, triggering specification, color space, number of acquired image at triggering, and region of interest while acquiring. This toolbox will bridge between frame grabber and MATLAB environment. Image Processing Toolbox provides many reference algorithms, graphical tools, analysis, etc. Reference algorithm provides fast development of algorithms. Filters, transforms, and enhancements are ready to use functions which simplify to code generation. This toolbox is used in face detection and some part of face recognition sections.

4.2 Coding Face Recognition With Matlab

Many in-built MATLAB functions are for image processing since it is a numerical computing environment that allows easy matrix manipulation. Matlab is used to test new image processing techniques and algorithms. Almost everything in MATLAB is done through programming and manipulation of raw image data. MATLAB does include standard “for” and “while” loops, but using MATLAB’s vectorized notation often produces code that is easier to read and faster to execute. In this, face recognition using singular value decomposition were recoded with MATLAB. The programs utilize matrices operation to manipulate data to reduce “for” loops, which reduce lines of the code.

The consideration for using matrix operations is that the inner matrix dimensions must agree. In the program below, U is a $M \times M$ right matrix of Singular Value Decomposition of $A(M \times N)$, Ur is $M \times r$ matrix that are form from U (r is the number of singular values we choose). $\mathbf{X} = Ur * A$ is the coordinates (position) matrix that is the $r \times N$ dimensions for training set A. $\mathbf{x}$ is a coordinates vector in the $r \times N$ subspace for testing image.

$$
\begin{aligned}
\mathbf{X} &= Ur * A \\
&= [Ur * a_1, Ur * a_2, ..., Ur * a_N] \\
&= [x_1, x_2, ..., x_N]
\end{aligned}
$$

Where

$$= \mathbf{x}_1 = \begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \\ \cdots \\ x_{1r} \end{bmatrix}, \mathbf{x}_2 = \begin{bmatrix} x_{21} \\ x_{22} \\ x_{23} \\ \cdots \\ x_{2r} \end{bmatrix}, \mathbf{x}_N = \begin{bmatrix} x_{N1} \\ x_{N2} \\ x_{N3} \\ \cdots \\ x_{Nr} \end{bmatrix}$$

and

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \cdots \\ x_r \end{bmatrix}$$

When computing the distance of each points between the training set and testing image,

$$D = \mathbf{X} - \mathbf{x} * ones(1, N)$$

In order to have agreed dimensions to operate the matrix $\mathbf{X}$ and vector $\mathbf{x}$, we need to transfer the vector $\mathbf{x}$ to the matrix form by operation $\mathbf{x} * ones(1, N)$,

$$\mathbf{x} * ones(1, N) = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \cdots \\ x_r \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \end{bmatrix}$$

$$= \begin{bmatrix} x_1 & x_1 & x_1 & \cdots & x_1 \\ x_2 & x_2 & x_2 & \cdots & x_2 \\ x_3 & x_3 & x_3 & \cdots & x_3 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_r & x_r & x_r & x_r & x_r \end{bmatrix}_{(r\times N)}$$

So that the coordinates distance matrix between the testing image and each image in the training set

$$D = [D_1, D_2, D_3..., D_N]$$

Where

$$D_1 = \begin{bmatrix} x_{11} - x_1 \\ x_{12} - x_2 \\ x_{13} - x_3 \\ \cdots \\ x_{1r} - x_r \end{bmatrix}, D_2 = \begin{bmatrix} x_{21} - x_1 \\ x_{22} - x_2 \\ x_{23} - x_3 \\ \cdots \\ x_{2r} - x_r \end{bmatrix}, ..., D_N = \begin{bmatrix} x_{N1} - x_1 \\ x_{N2} - x_2 \\ x_{N3} - x_3 \\ \cdots \\ x_{Nr} - x_r \end{bmatrix}$$

The minimized distance between training face and testing image:

$$d = sqrt(diag(D^T \times D))$$

where

$$D^T \times D = \begin{bmatrix} D_1^T \\ D_2^T \\ D_3^T \\ \cdots \\ D_N^T \end{bmatrix} \begin{bmatrix} D_1 & D_2 & D_3 & \cdots & D_N \end{bmatrix}$$

$$
= \begin{bmatrix} D_1^T D_1 & D_1^T D_2 & D_1^T D_3 & \cdots & D_1^T D_N \\ D_2^T D_1 & D_2^T D_2 & & & \\ D_3^T D_1 & & D_3^T D_3 & & \\ \cdots & & & \cdots & \\ D_N^T D_1 & & & & D_N^T D_N \end{bmatrix}
$$

$$
diag(D^T \times D) = \begin{bmatrix} D_1^T D_1 & D_2^T D2 & D_3^T & \cdots & D_N^T D_N \end{bmatrix}
$$

Therefore

$$
d = sqrt(diag(D^T \times D))
$$

is a vector of the minimized distance between the training face images and testing image. The source code is shown below.

```
function [ef, d] = svdRecognition0(newName, r, N, A, U, S, V,
fbar, e0, e1)
%newName = 'gideon.jpg'? r = number of singular values choosed;
Ur = U(:, 1:r); X = Ur'*A;
fid = fopen(newName);
face = fgetl(fid);
fnew = rgb2gray(imread(face));
fnew = imresize(fnew, [112, 92]);
f = reshape(fnew, 10304, 1);
f0 = double(f) - fbar;
x = Ur'*f0;
fp = Ur*x;
ef = norm(f0-fp);
if ef < e1
```

```
D = X-x*ones(1, N);
d = sqrt(diag(D'*D));
[dmin, indx] = min(d);
 if dmin < e0
   fprintf(['This image is face#', num2str(indx)]);
  else
fprintf('The input image is an unknown face');
 end
else
 d=0;
fprintf('The input image is not a face');
end


function [A, U, S, V, fbar] = svdRecognition(fileName, N)
%fileName = 'knust.txt';
fid = fopen(fileName);
chk = imread('Newface/valencia.jpg');
siz = size(chk);
S = zeros(siz(1)*siz(2), N);
  for i=1:N
      face = fgetl(fid);
      fi = rgb2gray(imread(face));
      subplot(ceil(sqrt(N)),ceil(sqrt(N)),i);
      fprintf(1,'%s.\n',face);
      figure(1); imshow(fi);
      fi = double(reshape(fi,siz(1)*siz(2),1));
```

```
        S(:,i) = fi;
    end
    S = S';
fbar = (mean(S))';
figure(2); imshow(reshape(uint8(fbar),112, 92));
A = S' - fbar*ones(1, N);
[U, S, V] = svd(A, 0);
```

4.3 Results for Face Recognition

The test was conducted under a training set with images of Size $M = 92 \times 112 = 10,304$,. The number of known individual face images is: $N = 40$, Different Conditions such all frontal and slight tilt of the head, different facial expressions and the use of glasses were taken into consideration. Essentially, a face image is of M (say 10,000) dimension. But the rank r of matrix A is less than or equals N. For most applications, a smaller number of basefaces than r are sufficient for identification. In this way, the amount of computation is greatly reduced. The following figures show the base face image, the average of training set image, and the training set image that was used for this experiment.

Figure 4.1: RGB Images of the Training Set

The face recognition in this study was performed using the RGB images in figure(4.1). This is possible because the recoded Matlab programme converted all images into grayscale before the implementation.

The figure(4.2) shows the image of the computed mean face of the training set images.

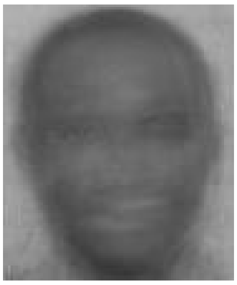

Figure 4.2: Grayscale Image of the Computed Mean face of the Training Set

4.4 Experiments on the Properties of SVD

One of the properties of the Singular Value Decomposition is that, the singular values $\sigma_1, \sigma_2, \sigma_3, ..., \sigma_n$ are unique, but the singular vectors U and V are not unique. The uniqueness of the singular values makes its study very important. To investigate this characteristic of the Singular Value Decomposition, an experiment was performed using three person's face images by combining their singular values and the singular vectors. The experiment was repeated using the face image of a person and a flower as a confirmation of the results obtained.

4.4.1 Experiments on the Properties of SVD using three Face images

The Singular Value Decomposition of the images were performed. The face images of Gideon (f_1), Dora (f_2) and Baby (f_3) were first decomposed into U, S, V^T so that $f_1 = U_1 S_1 V_1^T$, $f_2 = U_2 S_2 V_2^T$ and $f_3 = U_3 S_3 V_3^T$. The result of the combination of the singular values and singular vectors of the face images is shown in the table below.

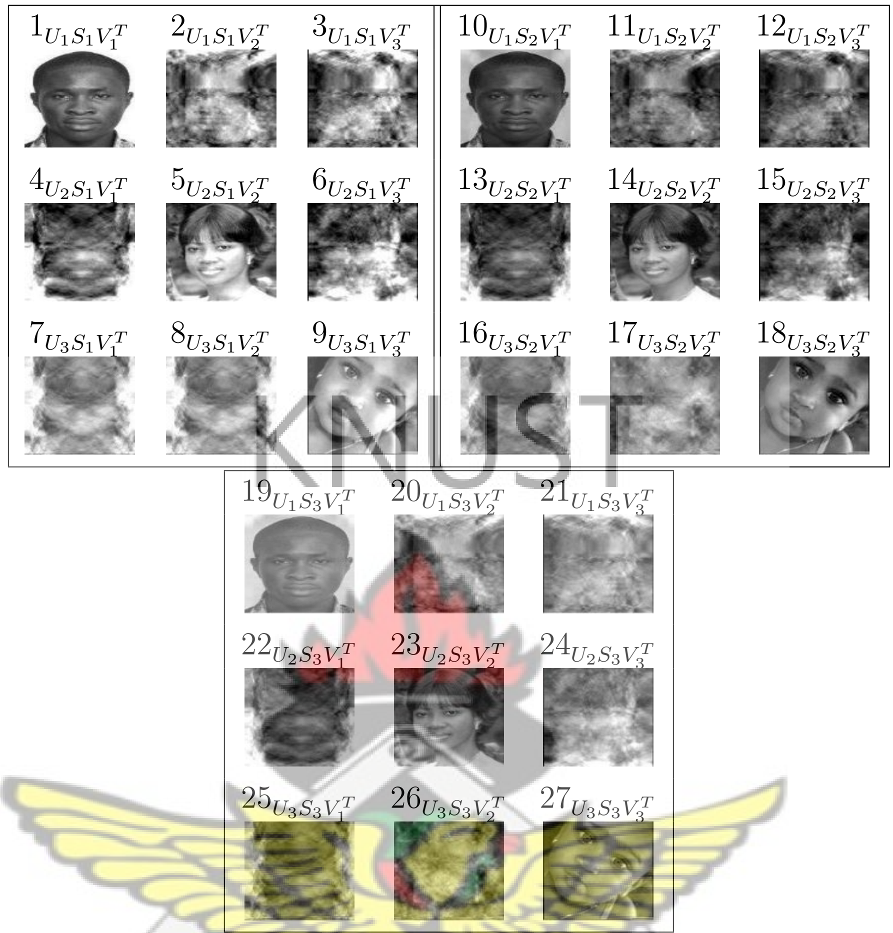

Table 4.1: Result of Exchanged Singular Values with Singular Vectors for three face images

In Table (4.1), the images labelled 1, 14 and 27, shows the combination of the singular values and singular vectors, $U_1S_1V_1^T$, $U_2S_2V_2^T$ and $U_3S_3V_3^T$ which are the original images of Gideon, Dora and baby respectively. However, when we combined Gideon's singular values (SVs) with Dora's singular vectors,(U_2) and (V_2^T) it shows Dora's face image (see Table(4.1) (Image. 5). The image has different brightness with Doras original image as shown in Table (4.1) Image 14. The difference in brightness is due to the fact that, the singular values are not from the face image of Dora. Image 9 shows Baby's face, but it is a combination of Gideon's singular values (SVs) and Baby's singular vectors(U_3) and (V_3^T). Image 10 shows Gideon's face, but it is a combination of Dora's singular values(S_2)

and Gideon's singular vectors(U_1) and V_1^T. Similarly, Image 18 shows Baby's face, but it is a combination of Dora's singular values (SVs) and Baby's singular vectors U_3 and V_3^T.

When we combined two pair of singular vectors U_i and V_j^T, which are from two different images i and j respectively, with the singular values of either image i or j, the outcome images looks like the "cloud". The results are shown in images 2, 3, 4, 6, 7, 8, 11, 12, 13, 15, 16, 17, 20, 21, 22, 24, 25 and 26 of Table (4.1).

The experiment showed that the combinations of Gideon's singular vectors(U_1, V_1^T), Dora's singular vectors(U_2, V_2^T), and Baby's singular vectors(U_3, V_3^T) with any singular value S_i, ($i = 1, 2, 3$) produces Gideon's image, Dora's image and Baby's image respectively. These images even though appear to be the same, the are numerically different. The images with the corresponding line graphs of the pixels (x, y) below, demonstrate clearly, these differences.

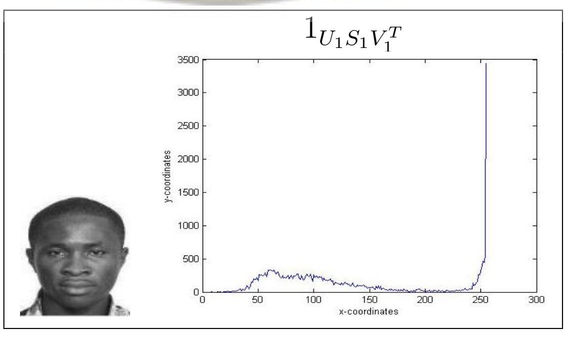

Table 4.2: A graph of Image with combination $U_1 S_1 V_1^T$

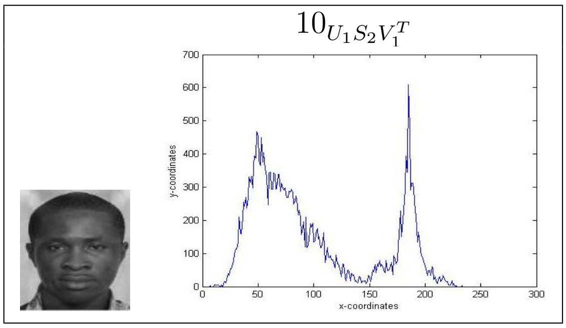

Table 4.3: A graph of Image with combination $U_1S_2V_1^T$

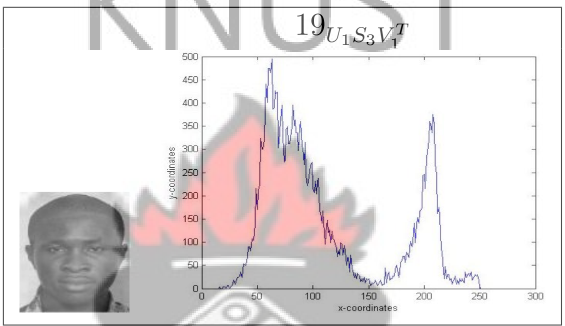

Table 4.4: A graph of Image with combination $U_1S_3V_1^T$

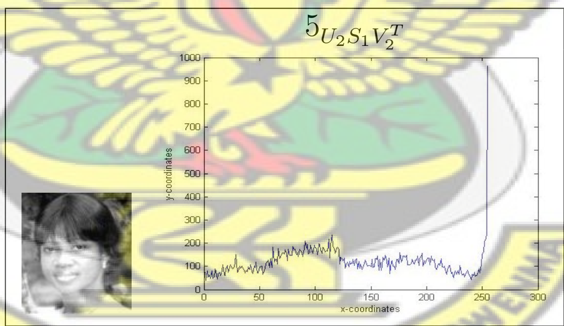

Table 4.5: A graph of Image with combination $U_2S_1V_2^T$

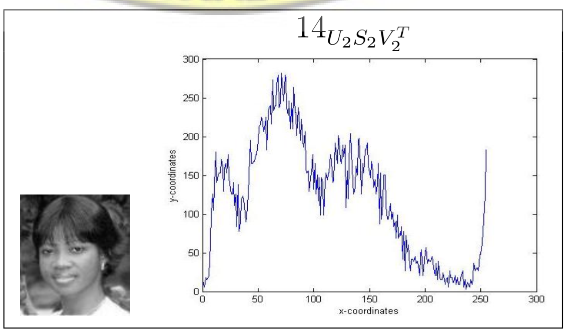

Table 4.6: A graph of Image with combination $U_2S_2V_2^T$

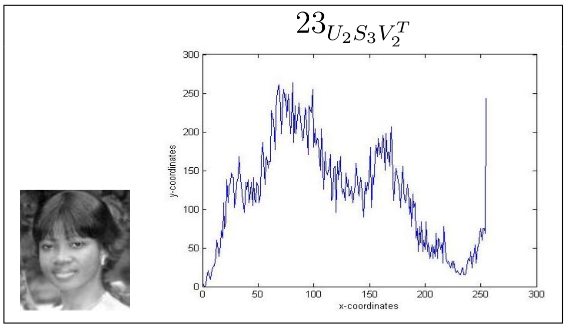

Table 4.7: A graph of Image with combination $U_2S_3V_2^T$

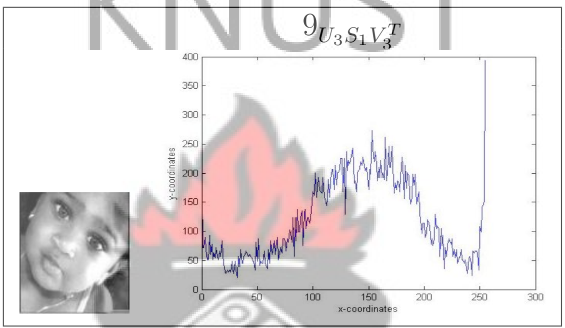

Table 4.8: A graph of Image with combination $U_3S_1V_3^T$

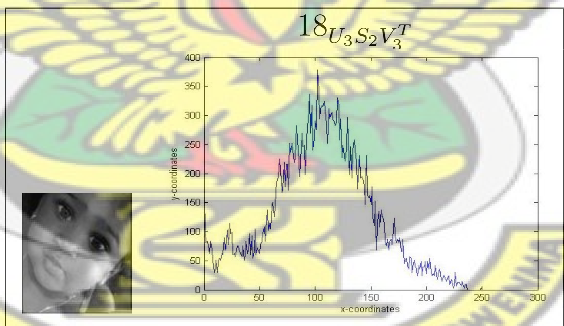

Table 4.9: A graph of Image with combination $U_3S_2V_3^T$

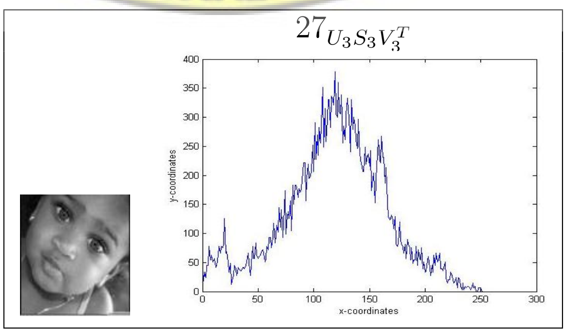

Table 4.10: A graph of Image with combination $U_3S_3V_3^T$

4.4.2 Experiments on the Properties of SVD using a Face Image and a Flower

The same experiment was performed using two images where one is a face image and the other, a flower image. Again, the singular value decomposition of the images were again performed. The result of the combination of the singular values and singular vectors of the face image and the flower is shown in the Table below.

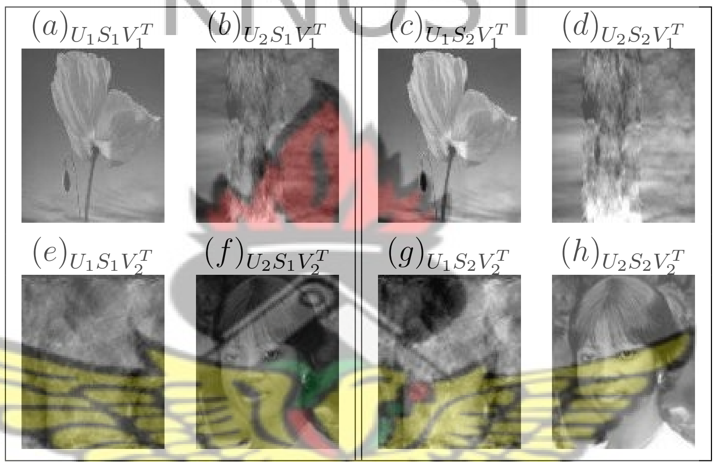

Table 4.11: Result of Exchanged Singular Values with Singular Vectors for a face image and a flower

The experimentation showed the same result as two face images. In Table(4.11), images (a) and (h) shows the combination of $U_1S_1V_1^T$, and $U_2S_2V_2^T$ which are the original images of the flower and Dora respectively. However, when we combined the singular values of the flower with Dora's singular vectors, it shows Dora's face image (see Table(4.11) image (f)). This image has different brightness when compared Dora's original image in (h). (c) shows the image of the flower, but it is a combination of Dora's singular values and the singular vectors of the flower. When we combined two pair of singular vectors U_i and V_j^T, from two images i and j respectively, the outcome images, again, looks like a "cloud". (See Table(4.11), images (b), (d) (e) and (g)).

The experiment as seen above, showed that the combinations of Dora's singular vectors(U_1, V_1^T), and the flower's singular vectors(U_2, V_2^T) with any singular value S_i, $(i = 1, 2)$ produces Dora's image and the flower's image respectively. These images are numerically different even though they appear to be the same. The images with the corresponding line graphs of the pixels (x, y) below demonstrate clearly, these differences.

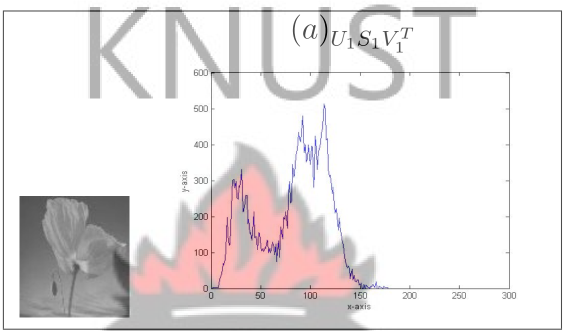

Table 4.12: A graph of Image with combination $U_1S_1V_1^T$

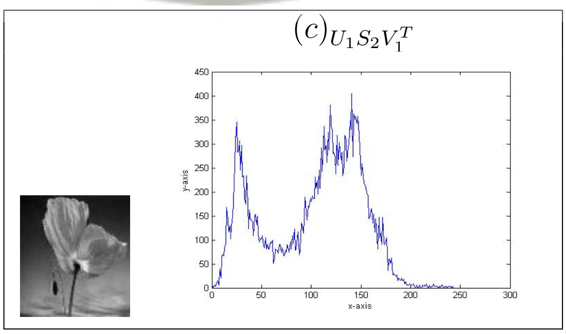

Table 4.13: A graph of Image with combination $U_1S_2V_1^T$

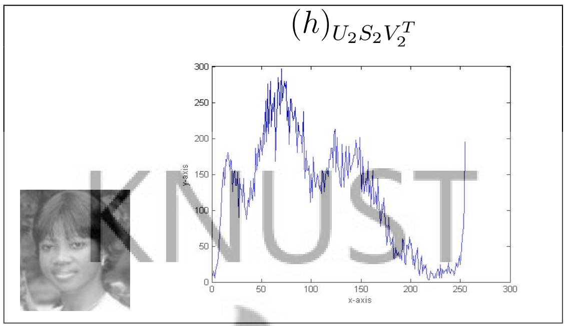

Table 4.14: A graph of Image with combination $U_2S_2V_2^T$

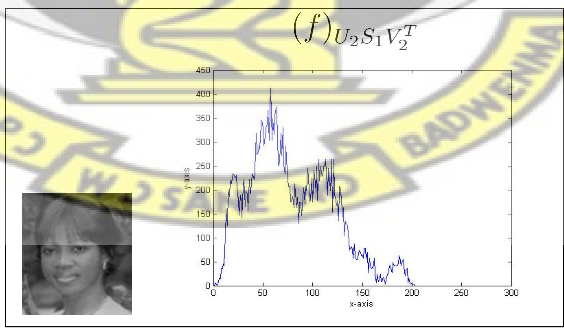

Table 4.15: A graph of Image with combination $U_2S_1V_2^T$

Chapter 5

Conclusion and Recommendation

5.1 Conclusion

The study first reviewed the concept of singular value decomposition. It then briefly mentioned some application areas such as the low ranked approximation of matrices, the pseudo-inverse of matrices and the solution to linear systems via singular value decomposition.

Singular value decomposition as a technique of linear algebra, has been applied to digital image processing. Face recognition as an area of image processing was investigated and tested. Based on the theory and result of experiments, it was found that singular value decomposition is a stable and effective method to decompose a system into a set of linearly independent components, each of them carrying its own data (information) to contribute to the system. That is, both rank of the problem and subspace orientation can be determined.

The face recognition test performed using the image that project into facebase show that it is necessary to improve the algorithm to work with complex objects. The singular value decomposition approach is robust, simple, easy and fast to implement. It works well in a constrained environment and provides a practical solution to image recognition problems. Instead of searching a large database of faces, by using basefaces, this small set of likely matches for given images can be easily obtained.

From the result it is clear that, though the singular values are unique in singular value decomposition, the singular vectors are more important for image processing. Especially, for face recognition. This fact necessitate a deep research and further investigation on the characteristics of singular value decomposition in image processing.

From the findings of this study, it is recommended that institutions that house a lot of people with different face images adapt the use of face recognition to enhance security.

5.1.1 Further Study

For further study, one may consider working on more complex images such as vary large size 3D images with singular value decomposition technique for image recognition. Also, the application can be performed with programming of Octave, Python or in some other Programming Languages to achieve real-time image processing.

References

Autonne, L. (1913). Sur les matrices hypohermitiennes et les unitairs. *Comptes Rendus de l'Academie des Sciences, Paris*, *156*, 858-860.

Bateman, H. (1908). A formula for the solving function of a certain integral of the second kind. *Transactions of the cambridge Philosophical Society*, *20*, 179-187.

Bau, D., & Trefthen, L. N. (1997). *Numerical linear algebra*. Society of Industrial and Applied Mathematics, Philadelphia.

Beltrami, E. (1990). *Sulle funzioni bilineari. giornale di matematiche ad uso degli stuenti delle universita* (Tech. Rep.). University of Minnesota, Department of Computer Science.

Benyah, F. (2012). Applications of singular value decomposition. In *The* 18^{th} *eward a. Bouchet/Abdus salam workshop.*

Blank, S., Krikorian, N., & Spring, D. (1989). A geometrically inspired proof of the singular value decomposition. *American Mathematical Monthly*, *96*, 238-239.

Bradley, G. L. (1975). *A primer of linear algebra* (A. Wester & H. Dekeukelaere, Eds.). Prentice-Hall, Inc. Eaglewood Cliffs, New Jersey.

Cheng, Y., Liu, K., Yang, J., & Wang, H. (1992). A robust algebraic method for human face recognition,. In *11th international confernce on pattern recognition.*

Datta, B. N. (1989). *Numerical linear algebra and its applications* (G. Ostedt, Ed.). Brooks/Cole Publishing Company.

Eckart, C., & Young, G. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, *1*, 211-218.

Eckart, C., & Young, G. (1939). A principal axis transformation for non-hermitian matrices. *Bulletin of the American Mathematics Society*, *45*, 118-121.

Gohberg, I. C., & Krein, M. G. (1969). Introduction to the theory of linear non-selfadjoint opeartors. *American Mathematical Society, Providence, Rhode Island*.

Golub, G., & Kahan, W. (1965). Calculating the singular values and pseudo-inverse of a matrix. *J. SIAM NUMER. ANAL*, *2*, 205-224.

Gonzalez, R. C., Woods, R. E., & Eddins, S. L. (2004). *Digital image processing using matlab.* Pearson Prentice-Hall, Upper Saddle River, New York.

Gray, R. M., & Neuhoff., D. L. (1998). Quantization. *IEEE Transactions on Information Theory*, *44*, 1-63.

Hasan, H. M., Jouhar, W. A. A., & Alwan, M. A. (2012). Face recognition using improved FFT based radon by pso and pca techniques. *International Journal of Image Processing (IJIP)*, *6*, 26-37.

Hong, Z. (1991). Algebraic feature extraction of image for recognition. , *24*, 211-219.

Jahne, B. (2002). *Digital image processing.* Springer-Verlag Berlin Heidelberg New York.

Jain, P., & Gautam, B. (2012). Performance estimation for face recognition using SVD. *International Journal of Engineering and Innovative Technology (IJEIT)*, *2*, 361-364.

Kachare, N. B., & Inamdar, V. S. (2010). Survey of face recognition tecchniques.

International Journal of Computer Applications (0975 - 8887), *1*, 30-34.

Lawrence, S., Giles, L., Tsoi, A. C., & Back, A. D. (1998). Face recognition: A convolutional neural network approach. In *Ieee trans. on neural networks, special issue on neural networks and pattern recognition.*

Leon, S. J. (1996). *Linear algebra with applications.* Macmillan Publishing Company, New York.

Meyer, C. D. (2000). *Matrix analysis and applied linear algebra.* SIAM.

Pagadala, S. (1998). *Applications of the singular value decomposition to image processing.* Unpublished master's thesis, Osmania University, Hyderadad, India.

Patil, A., Kolhe, S., & P.M, P. (2010). 2d face recognition techniques: A survey. *International Journal of Machine Intelligence*, *2*, 74-83.

Rudin, W. (1973). *Functional analysis* (J. L. Farnsworth, B. Bayne, & M. Margolies, Eds.). McGraw-Hill Book Company.

Shilov, G. E. (1977). *Linear algebra,.* Dover Publications, Mineola, NY.

Soumya, T., Soumya, V., & Soman, K. (2009). Singular value decomposition a classroom approach. *International Journal of Recent Trends in Engineering*, *1*, 1-4.

Stewart, G. W. (1992). On the early history of the singular value decompositions. *SIAM Review*, *35*(4), 551-566.

Strang, G. (1980). *Linear algebra and its applications.* Academic Press, New York, NY.

Sylvester, J. (1889a). A new proof that a general quadratic may be reduced to its canonical form (that is, a linear function of squares) by means of real orthogonal substitution. *Messenger of Mathematics*, *19*, 1-5.

Sylvester, J. (1889b). On the reduction of a bilinear quadratic of the n^{th} order to the form of sum of n products by a double orthogonal substitution. *Messenger of Mathematics*, *19*, 42-45.

V.J, R., & Dasgupta, A. (n.d.). Jpeg image compression using singular value decomposition. *International Journal of Advances in Electronics Engineering*.

Weyl, H. (1912). Das asymptotische verteilungsgesetz der eigenwert linearer partieller differentialgleichungen (mit einer anwendung auf therie der hohlrausmstrahlung). *Mathematsche Annalen*, *71*, 441-479.

Weyl, H. (1949). Inequalities between the two kinds of eigenvalues of a linear transformation. *Proceedings of the National Academy of Sciences*, *35*, 408-411.

Wikibooks Contributors. (2008). *LaTeX.* (Created on Wikibooks, the open content textbooks collection.)

Yong-Qing, C. (1991). Human face recognition method based on the statistical model of small sample size. *SPIE Proc: Intell. Robots and Computational Vision*, *1607*, 85-95.

Zeng, G. (2006). Face recognition with singular value decomposition. *CISSE Proceeding*.

Zhang, J., Yan, Y., & Lades, M. (1997). Face recognition: Eigenface, elastic matching, and neural nets. *IEEE*, *85*.

Appendix A

Matlab Source codes

A.1 Code for Experiments on the Properties of SVD

```
A1=imread('gideon.jpg');
A2=imread('dora.jpg');
A3=imread('baby.jpg');
s1 = size(A1); % find out the size of the image
s2 = size(A2);
s3 = size(A3);
ss1 =size(s1);
ss2 =size(s2);
ss3 =size(s3)
 if ss1(:,2)== 3; % if the image is a color image in jpeg or jog
                  %formatn it will covert to the greyscale
   A1 = rgb2gray(A1);
 end
```

```
 if ss2(:,2)== 3; % if the image is a color image in jpeg or jog
                  %formatn it will covert to the greyscale
  A2 = rgb2gray(A2);
 end
 if ss3(:,2)== 3; % if the image is a color image in jpeg or jog
                  %formatn it will covert to the greyscale
  A3 = rgb2gray(A3);
 end
[u1,s1,v1]=svd(double(A1));
[u2,s2,v2]=svd(double(A2));
[u3,s3,v3]=svd(double(A3));
combinf1=uint8(u1*s1*v1');
figure(1);imshow(combinf1);
title('combination of u1*s1*v1');
combinf2=uint8(u1*s1*v2');
figure(2); imshow(combinf2);
title('combination of u1*s1*v2');
combinf3=uint8(u1*s1*v3');
figure(3); imshow(combinf3);
title('combination of u1*s1*v3');
combinf4=uint8(u2*s1*v1');
figure(4); imshow(combinf4);
title('combination of u2*s1*v1');
combinf5=uint8(u2*s1*v2');
figure(5); imshow(combinf5);
title('combination of u2*s1*v2');
```

```
combinf6=uint8(u2*s1*v3');
figure(6); imshow(combinf6);
title('combination of u2*s1*v3');
combinf7=uint8(u3*s1*v1');
figure(7); imshow(combinf7);
title('combination of u3*s1*v1');
combinf8=uint8(u3*s1*v1');
figure(8); imshow(combinf8);
title('combination of u3*s1*v1');
combinf9=uint8(u3*s1*v3');
figure(9); imshow(combinf9)
title('combination of u3*s1*v3')
combinf10=uint8(u1*s2*v1');
figure(10); imshow(combinf10)
title('combination of u1*s2*v1')
combinf11=uint8(u1*s2*v2');
figure(11); imshow(combinf11)
title('combination of u1*s2*v2')
combinf12=uint8(u1*s2*v3');
figure(12); imshow(combinf12)
title('combination of u1*s2*v3')
combinf13=uint8(u2*s2*v1');
figure(13); imshow(combinf13)
title('combination of u2*s2*v1')
combinf14=uint8(u2*s2*v2');
figure(14); imshow(combinf14)
```

```
title('combination of u2*s2*v2')
combinf15=uint8(u2*s2*v3');
figure(15); imshow(combinf15)
title('combination of u2*s2*v3')
combinf16=uint8(u3*s2*v1');
figure(16); imshow(combinf16)
title('combination of u3*s2*v1')
combinf17=uint8(u3*s2*v2');
figure(17); imshow(combinf17)
title('combination of u3*s2*v2')
combinf18=uint8(u3*s2*v3');
figure(18); imshow(combinf18)
title('combination of u3*s2*v3')
combinf19=uint8(u1*s3*v1');
figure(19); imshow(combinf19)
title('combination of u1*s3*v1')
combinf20=uint8(u1*s3*v2');
figure(20); imshow(combinf20)
title('combination of u1*s3*v2')
combinf21=uint8(u1*s3*v3');
figure(21); imshow(combinf21)
title('combination of u1*s3*v3')
combinf22=uint8(u2*s3*v1');
figure(22); imshow(combinf22)
title('combination of u2*s3*v1')
combinf23=uint8(u2*s3*v2');
```

```
figure(23); imshow(combinf23)
title('combination of u2*s3*v2')
combinf24=uint8(u2*s3*v3');
figure(24); imshow(combinf24)
title('combination of u2*s3*v3')
combinf25=uint8(u3*s3*v1');
figure(25); imshow(combinf25)
title('combination of u3*s3*v1')
combinf26=uint8(u3*s3*v2');
figure(26); imshow(combinf26)
title('combination of u2*s2*v1')
combinf27=uint8(u3*s3*v3');
figure(27); imshow(combinf27)
title('combination of u3*s3*v3')
```