Adaptive Algorithm for Parameterization of Feature Extraction Techniques in Remote Sensing Images

By

EDMORE CHIKOHORA

(Student Number: 24744190)
BSc Mathematics & Computer Science, MSc Computer Science.

A Thesis Submitted in Fulfillment of the Requirements for the Award of the Degree of Doctor of Philosophy (PhD) in Computer Science

Department of Computer Science
School of Mathematics and Physical Sciences
Faculty of Agriculture, Science and Technology
North-West University, Mafikeng Campus
South Africa

Supervisor : Professor Obeten O. Ekabua
Co-Supervisor: Professor Bukohwo M. Esiefarienrhe

November 2015

Declaration

I declare that this research study on ***Adaptive Algorithm for Parameterization of Feature Extraction Techniques in Remote Sensing Images*** is my work, and has never been presented for the award of any degree in any university. All the information used has been duly acknowledged both in text and in the references.

Signature _____ Date _____

Chikohora, Edmore

Approval:

Signature: _____ Date _____

Supervisor: **Prof. O. O. Ekabua**
Department of Computer Science
Faculty of Agriculture Science and Technology
North-West University, Mafikeng Campus
South Africa

Signature: _____ Date 01/09/2016

Supervisor: **Prof. B. M. Esiefarienrhe**
Department of Computer Science
Faculty of Agriculture Science and Technology
North-West University, Mafikeng Campus
South Africa

DEDICATION

This thesis is dedicated to my wife and children for their endless love and constant support throughout my research and studies.

ACKNOWLEDGEMENTS

The words "thank you" is not enough to express my sincere gratitude to the almighty God for giving me wisdom and strength to make my PhD research work possible. Working on this thesis has been so inspiring, often exciting and sometimes challenging but always interesting and full of academic adventures. It has been made possible by many other people, who have supported me up to the end.

I wish to express my sincere gratitude to my supervisor. Thank you Professor Obeten O. Ekabua for the inspiration, patience, guidance, encouragement and support from the day I enrolled as a PhD student at North-West University. Your passion for my research work was just enough to inspire me throughout. Thank you also to my co-supervisor Professor Bukohwo Michael Esiefarienrhe for the guidance and support during this research work.

I appreciate the members of staff in the Department of Computer Science at North-West University Mafikeng campus, Doctor N. Gasela and Ms Nosipho Dladlu for the unwavering support you showed. You have made my time in the department much more enjoyable and I feel honoured to have been part of this department.

I further thank my colleagues and friends in the Department of Computer Science for their encouragement. Working together has been so inspiring and eye opening.

A special thanks to my wife Teressa and family for their emotional support and encouragement throughout this research work.

Last but not least, I would like to thank the journals that helped me publish part of my work and the North-West University's Institutional Research Support for their financial support throughout my study.

ABSTRACT

Remote Sensing Images (RSI) involves identification of a Region of Interest (RoI) from a suitable Feature Extraction Algorithm (FEA) that best extract the identified RoI, a process known as Dimensionality Reduction. This process can yield better results if the FET's parameter selection strategy specifies an optimum parameter value combination. As a result, parameterization plays an important role in the use of FET algorithms. Most research work on improving parameter selection strategies are based on empirical experiments, which are often characterized by short runs that in most cases result in invalid conclusions. Although empirical experiments have produced acceptable results, this strategy tends to bias the parameter distribution to a narrow range thereby reducing the capability to discriminate the modeled objects. Therefore, the research study presented a novel adaptive heuristic algorithm based on the Gabor Filter (GF) to generate useful solutions to optimization of parameter selection strategies for FET in RSI. The adaptive heuristic named "GenApp" implements a genetic approach that is premised on Genetic Algorithms (GA). The GenApp uses a 16bits random generator to generate phenotype values that are feed into a genAid simulator to generate genotypes values. GenApp was evaluated against existing algorithms namely the GF and the MGF for efficiency, fitness of value and the worst case scenarios. The results presented in our simulations show the GenApp algorithm having the best fitness value in five (5) random simulation runs. On the other hand, the GenApp posted the highest complexity values of 0.63 and 0.03 points above the existing algorithms. This lapse is attributed the initialization steps of the GenApp that screen the initial data of population. The screening process eliminates the worst-case scenario of the GenApp algorithm; hence this could be seen as a trade-off in its overall performance as compared to other existing algorithms that fail to screen the initial population.

DEFINITION OF TERMS

Remote sensing: The science of deriving information about a feature, an object, or a phenomenon from a distance by analyzing the energy reflected or emitted by the feature [*1*].

Genetic algorithms: An evolutionary algorithm which generates each individual from some encoded form known as a "chromosome" or "genome". Chromosomes are combined or mutated to breed new individuals.

Genetic programming: An extension of the genetic algorithm, a model for testing and selecting the best choice among a set of results, each represented by a string.

Wavelength: The distance between successive crests of a wave, especially points in an electromagnetic wave.

Pixels: (derived from "picture elements"), is the basic unit of programmable colour on a computer display or in a computer image.

Orientation: The relative physical position or direction of an object or phenomenon.

Phase offset: The initial angle of an electromagnetic wave at its origin.

Aspect ratio: An image projection attribute that describes the proportional relationship between the width of the image and its height.

Neural networks: Is a system of programs and data structures that approximates the operation of the human brain.

Feature extraction: Simplifying the amount of resources required to describe a large set of data accurately.

Empirical experiments: Experiments that are guided or depend upon experience or observation alone, without the use of any scientific method or theory.

Spatial: Relative ground sampling distance of one pixel.

Spectral: The number of electromagnetic regions sampled.

Mathematical morphology: A technique that deals with the mathematical theory of describing shapes using sets.

GenApp: Derived from "Genetic Approach", a novel algorithm that implements the adaptive principles of the natural biological genes implemented using genetic algorithms.

Key-figures: Important points or sections in an image that is being extracted using SIFT approach.

Image gradient: A directional change in the intensity or color in an image, used to extract information from images.

Orientation histograms: A feature descriptor used in image processing for the purpose of object detection by counting occurrences of gradient orientation in localized portions of an image.

LIST OF ACRONYMS

GA : Genetic Algorithms

GF : Gabor Filters

GFn : Gabor Function

GLCM : Grey Level Co-occurrence Matrix

GP : Genetic Programming

MM : Mathematical Morphology

MR : Multi-Resolution

RoI : Region of Interest

RSI : Remote Sensing Images

TM : Template Matching

FET : Feature Extraction Techniques

SIFT : Scale Invariant Feature Transform

GFr : Gaussian Filter

OCS : Open Close Sequence

ID : Information Datagram

PCA : Principal Component Analysis

SE : Structuring Element

AA : Adaptive Algorithms

SVM : Support Vector Machine

RR : Recognition Rate

FS : F-Score

LDA : Linear Discriminant Analysis

ICA : Independent Component Analysis

CBR : Case-Based Reasoning

PSNR : Pixel-Signal Noise Reduction

SURF : Speeded Up Robust Features

UC : Use Case

FEA : Feature Extraction Algorithms

MGF : Modified Gabor Filter

SLM : Smallest Local Minimum

HLM : Highest Local Maximum

LMS : Least Mean Square

MSE : Mean Square Error

LDB : Local Discriminant Bases

SLAM : Simultaneous Localization Mapping

RE : Requirement Elicitation

NASM: Network Adaptive Switching Median

PWMAD: Pixel-wise Morphological Anomaly Detector

OCS : Open-Close Sequence

GRF : Growth-Rate Function

EM : Efficiency Measure

RNG : Random Number Generation

RoI : Region of Interest

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

Chapter 1: Introduction

1.0 Introduction

Remote Sensing Images (RSI) employs various Feature Extraction Techniques (FET) such as the Gray Level Co-occurrence Matrix (GLCM), Traditional Gabor Filters (GF), Morphological Morphology (MM), Multi-Resolution (MR), Hough Transform (HT) and Scale Invariant Feature Transform (SIFT) among other techniques. These techniques are however known to use fixed parameters such as frequency, orientation, angle, direction and distance whose values are determined using empirical experiments or methods [*2*].

Although experimentation on parameters has produced some desired results, the use of Adaptive Algorithms (AAs) to determine parameter values can significantly enhance accuracy on the images obtained by FET in RSI.

The science of remote sensing has emerged as one of the most fascinating subjects over the past three decades. Earth observation from space through various remote sensing instruments has provided a vantage means of monitoring land surface dynamics, natural resources management and the overall state of the environment itself [*3*].

Remote sensing imagery finds its application spanning over quite a number of domains and these include mapping land-use and cover, agriculture, soils mapping, forestry, city planning, archaeological investigations, military observation and geomorphologic surveying among other uses [*4*]. For example, in the late 1970s and early I980s, a great research effort was focused on the use of multispectral images (obtained through remote sensing) for crop inventory and crop production. The Large Area Crop Inventory Experiment (LACIE) demonstrated the feasibility of utilizing satellite-based multispectral data for estimation of wheat production based on techniques that are still in use today by crop production forecasters in the USDA Foreign Agricultural Service [*5*].

While remote sensing data can consist of discrete point measurement or a profile along a flight path, in this research we are mostly interested in measurements over a two dimensional

spatial grid such as images. Remote sensing systems, particularly those deployed on satellites, provide a repetitive and consistent view of the earth that is invaluable to monitoring the earth system and the effect of human activities on the earth [*6*].

Our study falls under RSI processing which encompasses feature extraction. In feature extraction, there are many areas of specialization that can be looked into. For example, some algorithms have been designed to identify specific target objects while others focus more generally on say buildings or roads extractions and the extraction techniques for these different specializations can vary substantially.

In this study, we focused on developing a novel algorithm that determines parameter values for feature extraction techniques using the adaptive principles of the biological genes. The GF technique was considered as our control. As a result, a detailed analysis on the functionalities, limitations and cited examples of the GF technique is discussed as a background to our study.

1.1 Background Study

Remote sensing in our scope is seen as the science of deriving information about a feature, an object or a phenomenon from a distance by analyzing the energy reflected or emitted by the feature [*7*]. The main energy detected by remote sensing systems is electromagnetic energy. Remote sensing uses sensors to measure the amount of electromagnetic energy exiting an object or a geographic area.

The sensors are characterized by different resolutions such as spatial, spectral, radiometric and temporal. It can be noted that objects and features on the earth's surface interact differently with the electromagnetic energy based on their molecular composition and the differences in the amount and properties of electromagnetic radiation becomes a valuable source of information.

Daugman [*4*], discussed the GF as a technique that implements one or multiple convolutions of an input image with a two dimensional Gabor Function (GFn). The filter has a real and an imaginary component representing orthogonal directions as illustrated in equations (1), (2) and (3) representing complex function, real part and imaginary part respectively.

$$g(x,y;\theta,\psi,\sigma,\gamma,\lambda) = exp\left(\frac{x'^2+\gamma^2y'^2}{2\sigma^2}\right) exp\left(i\left(2\pi\frac{x'}{\lambda}+\psi\right)\right) \quad (1)$$

$$Re\ g(x,y;\theta,\psi,\sigma,\gamma,\lambda) = exp\left(\frac{x'^2+\gamma^2y'^2}{2\sigma^2}\right) cos\left(2\pi\frac{x'}{\lambda}+\psi\right) \quad (2)$$

$$Im\ g(x,y;\theta,\psi,\sigma,\gamma,\lambda) = exp\left(\frac{x'^2+\gamma^2y'^2}{2\sigma^2}\right) sin\left(2\pi\frac{x'}{\lambda}+\psi\right) \quad (3)$$

where, $x' = xcos\theta + ysin\theta,\ y' = -xsin\theta + ycos\theta.$

In this equation, λ represents the wavelength of the sinusoidal factor, θ represents the orientation of the normal to the parallel stripes of a Gabor function, ψ is the phase offset, σ is the sigma of the Gaussian envelope and γ is the spatial aspect ratio. Bandwidth b is as well considered and specified as a real positive valued parameter as follows;

$$b = log_2 \frac{\frac{\sigma}{\lambda}\pi+\sqrt{\frac{ln2}{2}}}{\frac{\sigma}{\lambda}\pi-\sqrt{\frac{ln2}{2}}} \quad \text{and} \quad \frac{\sigma}{\lambda} = \frac{1}{\pi}\sqrt{\frac{ln2}{2}}\cdot\frac{2^b+1}{2^b-1} \quad (4)$$

Using the GFn it's evident that any slight changes made to the input values of the parameters λ, θ, ψ and γ has a huge effect on the output image. Petrov et al. [*8*], illustrated the direct effects of the parameters in equations (2) and (3) on the output image. We performed number of simulations with Figure 1.1 as input image to test the effect of changes in parameter values to the output image.

The following is a summary of the experiments performed using MatLab simulations of a Gabor Filter. A hexagon shape, Figure1.1 was randomly selected used as the input image representing a RoI ready to be extracted using GF extraction technique. The idea here is to illustrate the effects of parameter values on output from FET. Figures 1.2 to 1.4 show the effects of each incorrect parameter to the whole output image as indicated in each simulation. The results obtained confirmed the need for a proper parameter value combination when applying FET in RSI, hence the need for developing a parameter value selection strategy.

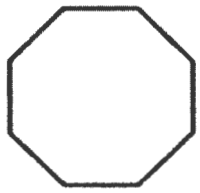


Figure 1.1: Input image for experiments 1, 2 & 3 [9]

Simulation 1: Testing the effects of Wavelength λ parameter on output image

The images of size 100 x 100 in Figure 1.2 shows GF kernels with values of the wavelength parameter of 5, 10 and 15, from left to right, respectively. The values of the other parameters were as follows: orientation 0, phase offset 0, aspect ratio 0.5, and bandwidth 1.

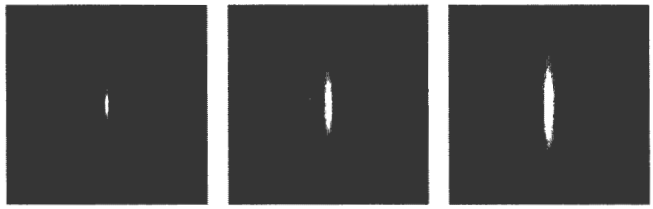


Figure 1.2: Output from Experiment 1 given Figure 1 as input image

Simulation 2: Testing the effects of orientation θ parameter on output image

The images of size 100 x 100 in Figure 1.3 shows GF kernels with values of the orientation parameter of 0, 45 and 90, from left to right, respectively. The values of the other parameters are as follows: wavelength 10, phase offset 0, aspect ratio 0.5, and bandwidth 1.

Figure 1.3: Output from Experiment 2 given figure 1 as Input Image

Simulation 3: Using empirical knowledge to determine parameter values

The image of size 100 x 100 in Figure 1.4 show GF kernels with the following parameter values; orientation 0, wavelength 8, phase offset 0, aspect ratio 0.5, and bandwidth 1. The output image, Figure 1.4, was reached at through experience, that is, after several trials on the parameter values.

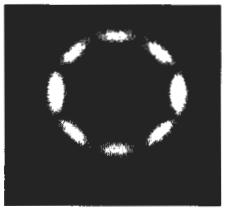


Figure 1.4: Output from Experiment 3 given Figure 1 as Input Image

1.2 Problem Statement and Research Questions

The problem statement and research questions are as discussed below.

1.2.1 Problem Statement

Parameter selection plays a crucial role in the use of FET such as the GF, MGF, MM and the SIFT. This has long been a research focus in the field of image processing.

Moreno et al. [*10*], worked on improving the GF parameter selection for local feature detection. Their strategy extended the Information Datagram whereby they analyzed the Gabor responds function on the parameters σ, γ and θ in view of selecting those parameters that best describe the particular object's characteristics. Their proposed strategy was a success in picking the best parameters semi-automatically from a set of parameters obtained through empirical experiments.

In video stabilization Santhaseelan et al. in [*11*] proposed a technique that automatically determine the variance for the Gaussian Filter (GFr) from the inter-frame transformations. In their approach named *"Adaptive parameterized filtering"*, they came up with a method that

estimates the standard deviation σ of the GFr by mapping the logarithmic function of the variation in translations as:

$$\sigma = \log((1 + \sigma_x^2)(1 + \sigma_y^2)), \quad (5)$$

Where σ_x and σ_y are the variance of the translation components in the X and Y directions respectively. Though their method was found to work well, it was based on observations from various experiments performed with regard to the components of the correlation matrix [*11*].

Most research papers we reviewed worked on improving parameter selection strategy for the GF and the SIFT which are mostly based on empirical experiments. Although the GF and the SIFT have produced acceptable results using parameters obtained through empirical experiments [*2*] [*11*], this strategy could bias the parameters distribution to a narrow range and reduce the capability to discriminate the modeled object(s) [*10*].

Empirical experiments from our experience are too often characterized by short runs which in most cases result in invalid conclusions. For example, if one flips a coin 5 times and gets 4 heads, we might assume that he/she will get heads 80% all the time, but in reality if you flip the coin millions of times, you'd get closer to 50% (variations would depend on how the coin is weighted).

The results from the simulations performed (simulations 1-4) show the effects of changes in parameter values to the output image. Figures 1.2 and 1.3 show how blurred the image could look when a FET fails to obtain optimum values for its parameters. Figure 1.4 show some improvement on the output image as compared to previous results, hence the need for an algorithm that specify parameter values for FET more consistently.

In our research, we worked towards optimizing the results obtained through FET in RSI by shifting from the use of empirical experiments in determining parameters and their values to using a novel adaptive algorithm that we named *"The GenApp"* derived from "Genetic Approach" that implement the adaptive principles of the natural biological genes that we implemented using GA.

1.2.2 Research Questions (RQ)

In consideration of the challenges faced by FET with fixed parameters in RSI processing, our research answered the following questions:

RQ1: How can we improve parameter selection strategy for FET in RSI processing?

RQ2: Is it possible for adaptive parameterization of FET to improve quality of images obtained in RSI processing?

RQ3: How can we develop an algorithm that implements adaptive parameterization for FET in RSI processing?

RQ4: As proof of concept, can we implement the adaptive algorithm through simulation and compare the results obtained with the existing approaches?

1.3 Research Rationale

RSI is the only practical way to obtain data from inaccessible regions such as the Amazon and feature extraction becomes the first and most crucial step towards obtaining information from such areas. The purpose of this study is to initially survey the characteristics of FET in RSI that are currently in use with particular attention to how such techniques specify their parameters. This was followed by the development of an algorithm that specifies parameters for FET in RSI processing adaptively.

On the other hand, this study helped researchers to learn alternative approaches that can help improve feature extraction in the field of RSI processing such as the application of GAs in solving RSI processing related problems.

1.4 Research Goal and Objectives

Our research goal and objectives are as discussed below.

1.4.1 Research Goal

The main goal of this research is to develop a novel adaptive algorithm based on the principles of natural biological genes that enhances parameterization of feature extraction in RSI.

1.4.2 Research Objectives (RO)

In order to achieve our research goal, we explored the following objectives:

RO1: To survey the existing literature for the characteristics of currently used FET in RSI processing such as the GF, GLCM and MM.

RO2: To perform requirements elicitation/ analysis for dynamic parameterization of FET in RSI processing.

RO3: To develop a novel adaptive algorithm that specifies the parameters for FET in RSI processing adaptively.

RO4: As proof of concept, to perform an evaluation of the proposed algorithm for parameterization of FET using a simulation tool.

1.5 Research Contributions to Knowledge (RC)

The main contribution of this research work to the research literature and RSI processing is the development of a novel adaptive algorithm that among other contributions:

RC1: Specify the parameters values for FET in RSI processing adaptively in such a way that enhances the quality of the output image by taking into consideration various obstacles, such as noise that could be associated with the RoI.

RC2: Help in the reconstruction of an image by providing an improved image detail and accuracy obtained through the use of FET in RSI processing.

RC3: Enabled an image-independent parameter selection scheme which is more flexible to the environment and the extraction technique. This should help in situations where FETs are design for a specific problem and environment.

1.6 Research Methodology

Our research methodology required that we initially gather relevant data from the specified documents and compiled databases that helped us to analyze the results sets obtained from different FET that use fixed parameters. This was followed by conducting simulations using genAID, MatLab and Microsoft Excel which helped us to address research questions and objectives discussed in sections 1.2.2 and 1.4.2 respectively.

The design followed a triangulation approach, where we made a comparative analysis of different results sets. That is, results obtained from FET that employ fixed parameters such as the GF and MGF with those obtained by our novel algorithm.

1.6.1 Literature Survey

Our literature survey addressed research question 1.2.2 (RQ1) and objective 1.4.2 (RO1) by way of reviewing selected journals, conference papers and other related publications on FET and RSI processing. This followed an in-depth analysis on the characteristics and operations of other FET such as the GF and MGF as well as their strategies for obtaining parameter values.

1.6.2 Dynamic Parameterization of Feature Extraction Techniques

The development of the new adaptive algorithm required that we first elicit and analyze its requirements. As a result, we performed a requirements elicitation for dynamic systems in RSI processing through a series of experiments and a case study on some randomly selected data using MatLab. This helped to determine a more effective requirements analysis technique that can be used in the development of dynamic systems such as the GenApp. This helped to answer research question 1.2.2 (RQ2) and objective 1.4.2 (RO2).

1.6.3 Algorithm Development

In order to develop the GenApp algorithm, we considered a tool that offered integrated capabilities for deep and broad exploration of algorithm design options, as well as efficient deployment to desktop and embedded software environments. The tool successfully simulated the commonly used genetic operators such as Population, Fitness function, Crossover, Mutation and Selection thereby addressing research question 1.2.2 (RQ3) and objective 1.4.2 (RO3).

1.6.4 Proof of Concept

As proof of concept, simulations on the proposed novel algorithm for parameterization were performed. We applied the *"Big Oh Notation"* and the Fitness Value test, algorithm evaluation techniques that evaluated the efficiency and fitness of the GenApp algorithm. The evaluations we did answered the requirements of research question 1.2.2 (RQ4) and objective and 1.4.2 (RO4).

1.7 Included and Related Publications

Our approach was to publish each of the core chapters of the thesis in peer-reviewed journals and here we present the chapters that have been published, under review and work in progress. The manuscripts included are as follows;

i. E. Chikohora and O. O Ekabua, "Feature Extraction Techniques in Remote Sensing Images: A survey on Algorithms, Parameterization and Performance," *International Journal of Soft Computing and Engineering,* vol. 4, no. 1, pp. 140-144, March 2014. (Chapter 2). This article is part of chapter two of our research work, we as well presented at North-West University, Mafikeng Campus, Faculty of Agriculture, Science and Technology Research Day, 08 October 2013.

ii. E. Chikohora, O. Ekabua, "A Requirements Elicitation Approach to Performing Dynamic Parameterization of Feature Extraction Algorithms," This paper form part of chapter three of our research work and is still work in progress.

iii. E. Chikohora and O. O. Ekabua, "A Genetic Approach to Parameterization of Feature Extraction Algorithms in Remote Sensing Images," *International Journal of Soft Computing And Engineering.*, vol. 4, no. 2, pp. 144-149, May 2014. This paper contributed towards chapter four of this research work.

iv. E. Chikohora, O. Ekabua, (2015) Analysis and Performance Evaluation of Feature Extraction Algorithms in Remote Sensing Images. IETE Technical Review, article reviewed and accepted for publication. The article forms part of the evaluation of the experiments we did in chapter five of this research work.

1.8 Thesis Outline

The outline of the rest of the thesis is as follows:

Chapter 2: Review of related literature. Cover a comprehensive review of related literature as discussed by other researchers covering FET and application of genetic algorithms in the field of RSI.

Chapter 3: Dynamic parameterization of FET. Outlines the different approaches used to do requirements elicitation in dynamic systems such as the GenApp.

Chapter 4: The GenApp: Novel adaptive algorithm. Provide a description of the development process of our proposed approach "the GenApp" for parameter determination in FET.

Chapter 5: Simulation and performance evaluation. In this chapter the developed novel algorithm is put to test by way of simulations and evaluations using carefully selected algorithms evaluation methods, such as the Big Oh Notation and Fitness Value tests.

Chapter 6: Summary, conclusion and future work, provides a summary of the thesis, its contributions, limitations and future directions in our research area.

Chapter 2: Review of Related Literature

2.0 Chapter Overview

In this chapter we looked at two major aspects of existing literature relevant to this research, which are FET and GA. In this regard, we divided the chapter into two major sections which we addressed separately as follows;

i. A critical and comprehensive review of the algorithms implemented by different FET and their parameter selection strategies.
ii. Application of genetic algorithms in RSI processing, with particular focus on feature extraction techniques.

The first section was further divided into three subsections where we initially selected and analyzed some of the widely used FET and the algorithms they implemented. Secondly, we surveyed their parameter selection strategies and finally made a critical analysis on their performance. Our analysis was mainly based on the literature results obtained from different publications we selected. This culminated in us making some concluding remarks on the success and limitations of the FET.

2.1 Preamble

Feature Extraction (FE) as discussed by Kumar et al., plays a very important role in the area of image processing [*12*]. In their research they further elaborate that before we get image features, various image preprocessing techniques like binarization, thresholding, resizing, normalization etc. are applied on the sampled image and feature extraction techniques are later applied after the preprocessing phases in order to get features that will be useful in classification and recognition of images.

The main idea behind FE is to obtain the most relevant information from the original data and represent it in a lower dimensionality space. If for example, an algorithm is too large to be processed and it is suspected to be redundant in such a way that it provides too much data but

less information, then the input data will be transformed into a reduced representation set of features named features vector. This transformation is what we call FE.

2.2 Feature Extraction

FE finds its application in many systems such as image classification, character and pattern recognition systems. In pattern recognition for example, input pattern is assigned to some possible output classes that correspond to the given input class. This process is usually done in two phases which are feature selection and classification. In these systems feature selection plays an important role in helping the classifiers to recognize features, given that most classifiers are not intelligent enough to recognize patterns from poorly selected features. As a result of this, the selection process should ensure that features have certain attributes to simplify the classification process such as containing required information to distinguish them between classes, insensitivity to irrelevant variability in the input classes and relevance to the output patterns to allow efficient computation of discriminant functions thereby reducing the required training data.

During pattern classification relevant information that characterizes each class is extracted from objects to form feature vectors that are used by classifiers to recognize the input with target output units [*12*]. FE becomes an important step in the construction of any pattern classification in that it finds the set of parameters that precisely and uniquely defines the shape using its feature vectors. This process helps in maximizing the recognition rate patterns using a minimum possible number of elements. In the late '80s researchers made attempts to automating the process of feature extraction which to some extent was a milestone achieved.

2.3 Feature Extraction History

Kern discussed automation of the feature extraction process as the *"Holy Grail"* of the photogrammetric data collection industry for many years [*13*]. The trade-off between the difficulty and the major benefits in automation of feature extraction process was the sole reason Kern in his publication labeled automation of feature extraction as the Holy Grail.

Works on automation of feature extraction started in the late 1980s with automated paper map conversion known as rasterization being popular. Although both automated and semi-automated methods were in use by then, the automated methods proved to be efficient at line rasterization as compared to text rasterizing. The success rate of feature extraction automation was measured at about 80% but the major challenge was that hindered its rapid progress and use was the time this process took to correct improperly rasterized text. This challenge led to the introduction of semi-automated software that would prompt the user to enter text as input. Semi-automated feature extraction process was bit robust and much faster in processing as compared to the automated process which required much more time to correct processing errors.

A breakthrough in automating feature was realized in 1990 when Digital Terrain Model collection from stereo pairs was achieved in digital photogrammetry. This was used in image processing to satellite imagery and produce GIS data using supervised and unsupervised image classification for small scale image data. It was after this breakthrough that many ideas came into fruition, such as the development of different automated algorithms for feature extraction. For example the "SNAKES" was concluded with a well-designed digital menu that was far much better and productive.

2.4 Classification of Feature Extraction

Kumar et al., discussed feature extraction and some of the approaches used to extract feature points known as FET [*12*]. In their publication, they classified these techniques into three major categories based on their approaches to dimensionality reduction. The categories are as discussed below:

2.4.1 Statistical Features

These features are derived from the statistical distribution of points and they are commonly known to provide high speed and low complexity while taking care to some extent feature style variations. Statistical features may also be used for reducing the dimension of the feature set (dimensionality reduction).

The following are the major statistical features:

(i) Zoning: The frame containing the character is divided into several overlapping or non-overlapping zones. Densities of the points and some strokes that are placed in different regions are analyzed of which they form the features. Contour direction features are used to measure the direction of the contours of the characters such as bending points. Bending points are points at which a stroke in the image has a strong curvature.

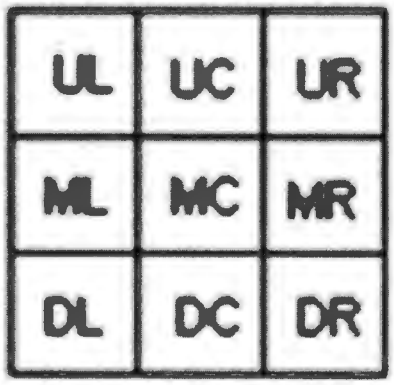


Figure 2.1: Matrix Representation of a Statistical Feature [*12*].

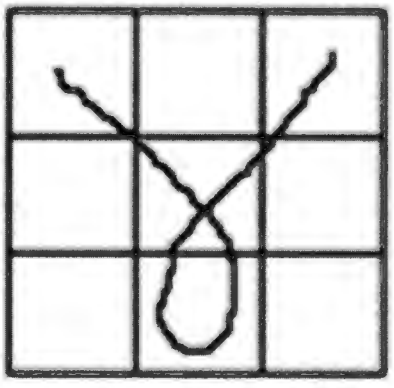


Figure 2.2: Non Uniform Statistical Feature [*12*].

(ii) Characteristic Loci: For every white point in the background of the character, vertical and horizontal vectors are generated. The number of times that the line segments intersected by these vectors are used as features.

(iii) Crossing and Distances: Crossing counts the number of transitions from background to foreground pixels along vertical and horizontal lines through the character image, while Distances calculate the distances of the first image pixel detected from the upper and lower boundaries of the image along the horizontal lines.

2.4.2 Global Transformation and Series Expansion Features

These features are invariant to global deformations like translation and rotations. A continuous signal generally contains more information that needs to be represented for the purposes of classification. One way to represent a signal is by a linear combination of a series of simpler well defined functions. The coefficients of the linear combination provide a compact encoding known as series expansion.

The following are some common transform and series expansion features:

(i) *Fourier Transforms:* The general procedure is to choose magnitude spectrum of the measurement vector as the features in a n dimensional Euclidean space. One of the most attractive properties of the Fourier Transform is the ability to recognize the position shifted characters when it observes the magnitude spectrum and ignores the phase. Fourier Transforms has been applied to Optical Character Recognition (OCR) in many ways.

(ii) *Walsh Hadamard Transform:* This feature is more suitable in high speed processing since the arithmetic computation involves only addition and subtraction. The major drawback of Walsh Hadamard transform is that its performance depends heavily upon the position of the characters.

(iii) *Rapid transform:* It is same as the Hadamard Transform except for the absolute value operation which may be credited with the elimination of the position shifting problem.

(iv) *Hough Transform:* It is a technique for baseline detection in documents. It is also applied to characterize parameter curves of characters.

(v) *Gabor Transform:* The Gabor transform is a variation of the windowed Fourier Transform. In this case the window used is not a discrete size but is defined by a Gaussian function.

(vi) *Wavelets:* Wavelet transformation is a series expansion technique that allows us to represent the signal at different levels of resolution.

(vii) *Karhunen Loeve Expansion:* It is an Eigen vector analysis which attempts to reduce the dimension of the feature set by creating new features that are linear combinations of the original features.

(viii) *Moments:* Moment normalization strives to make the process of recognizing an object in an image size.

2.4.3. Geometrical and Topological Features

These features may represent global and local properties of characters and have high tolerances to distortions and style variations. These topological features may encode some knowledge about the contour of the object or may require some knowledge as to what sort of components make up that object.

(i) *Strokes:* These are the primitive elements which make up a character. The strokes can be represented as lines and arcs which are the main strokes of Latin characters and can sometimes be as complex as curves and splines making up Arabic characters. In one line character recognition a stroke can also be defined as a line segment from pen down to pen up as illustrated in figure 2.3.

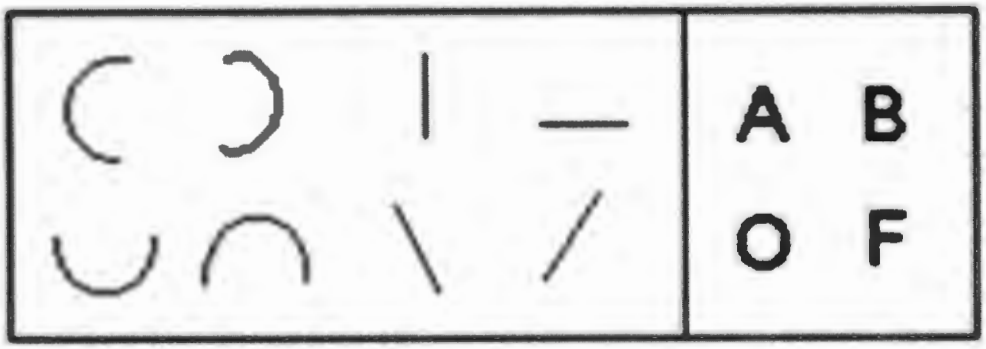


Figure 2.3: Main Strokes used for Latin Characters [*12*]

(ii) Stroke Directions and Bays: The sequence of directions of pen motion during the writing of a character is used as features.

(iii)Chain Codes: This feature is obtained by mapping the strokes of a character into a dimensional parameter space which is made up of codes as shown in figure 2.4.

(iv) End points intersections of line segments and loops.

(v) Strokes relations and angular properties.

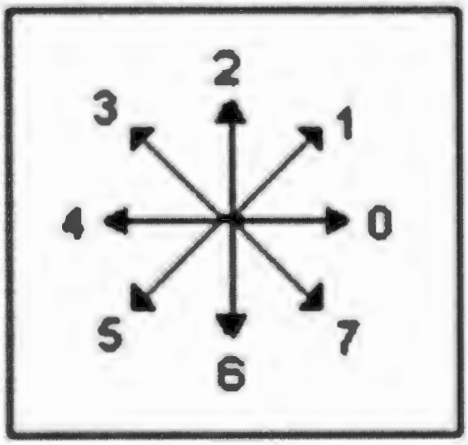


Figure 2.4: Chain Codes [*12*].

2.5 Overview of Different Feature Extraction Techniques

In machine learning, pattern recognition and image processing, features extraction performs some transformations of original features extracted from a source image to generate other features that are more significant to human interpretation. These transformations are performed by algorithms or methods named feature extraction techniques.

Here we provide an overview and classification of feature extraction techniques used to extract features from the original hyperspectral data. Since most of the algorithms are well known in remote sensing images literature, only a brief description of the concepts underlying the techniques is provided.

2.5.1 Classification of Feature Extraction Techniques

Feature extraction can be performed using various mathematical models, image processing techniques and computational intelligent tools such as neural networks or fuzzy logic [*14*]. Generally these techniques are classified into four categories namely, feature based, appearance based, template-based and part-based approaches. The last category is quite a new approach that finds its application mostly in recent computer vision and object recognition domain. Figure 2.5 illustrates the classifications of various feature extraction techniques.

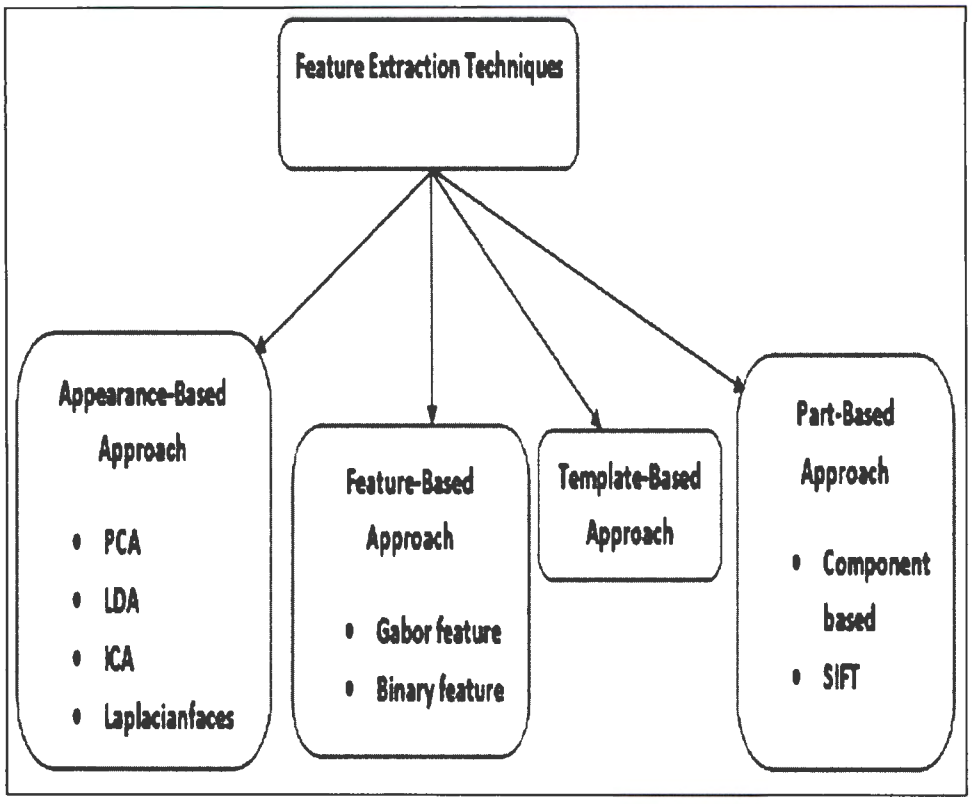


Figure 2.5: Classification of Feature Extraction Techniques [*14*].

2.5.1.1 Appearance- Based Approaches

In some literature, these approaches are referred to as holistic-based methods owing to their attempt to initially identify regions of interest using the entire image in place of local features and later perform some transformation on this patch to derive a compact representation for recognition. The transformations on the whole patch are performed based on information from statistical learning and analysis which are iterated until a feature is reached.

The main advantage of the appearance-based approaches is that they do not demolish any of the information contained in the images by focusing on only restricted regions or points of interest, however their major drawback is that they start out with the basic assumption that all the pixels in the image are have equal importance [*15*]. The PCA, LDA and ICA are examples of such approaches.

2.5.1.2 Feature Based Approaches

Feature based methods derive information directly from some detected fiducial points like eyes, noses, and lips, etc. [***14***], which are usually determined from domain knowledge, image processing, computer vision and are used to discard other irrelevant information. The Gabor and binary features are some of the examples of the feature based approaches.

The major advantage of feature based approaches is that they are relatively robust to position variations in the input image resulting from preprocessing activities such as image analysis and matching to some control data [***16***]. Other benefits include high speed matching and compactness of representation of images such as face features. However their main drawback is that they require user intervention in making decisions on which features return and/ or discard.

2.5.1.3 Template Based Approaches

Traditional template-matching approaches used distance metric for face recognition. Using this approach requires a set of symbolic templates for each class (person) and the similarity measurement is computed between a test image and each class. The class with the highest similarity score is selected as the correct match [***14***] and this is determined by using the Mahalanobis distance measure to compare the input image with the one already in the database. In cases where the results were not satisfactory the LDA was applied for further dimension reduction and classification. One example of such an approach is the Hough Transform.

2.5.1.4 Part Based Approaches

The part-based methods detect significant parts from the image, say a human face and combine the part appearances with machine learning tools for recognition. There is a slight difference with feature base approaches in that the later extracts the whole image (such as the whole face) and compare it to achieve the recognition purpose. Examples of such approaches are the component based and the Scale Invariant Feature Transform technique.

2.5.2 Traditional Gabor Filter

Daugman discussed the Traditional Gabor Filter (TGF) as a technique that implements one or multiple convolutions of an input image with a 2-D Gabor Function [*4*]. The function was viewed by Yang et al., [*17*] as a harmonic oscillator composed of sinusoidal plane wave of a particular frequency and orientation within a Gaussian envelope. The TGF views a complex 2-D filter over the image domain (x, y) as;

$$G(x,y) = \exp\left(\frac{(x-x_0)^2}{2\sigma_x^2} - \frac{(y-y_0)^2}{2\sigma_y^2}\right) exp(-2\pi i(u_0(x-x_0) + v_0(y-y_0))) \quad (6)$$

where (x_0, y_0) specify the location in the image (u_0, v_0), that has spatial–frequency $w_0 = \sqrt{u_0^2 + v_0^2}$ and orientation $\emptyset = \arctan\left(\frac{u_0}{v_0}\right)$. While σ_x and σ_y are the standard deviations of the Gaussian envelope along the x-axis and y-axis respectively. The real part of the filter is specified based on equation (6) as follows;

$$Re\ g(x,y;T,\emptyset) = \exp\left(-\frac{1}{2}\left[\frac{x_\emptyset^2}{\sigma_x^2} + \frac{y_\emptyset^2}{\sigma_y^2}\right]\right) cos\left(\frac{2\pi x_\emptyset}{T}\right) \quad (7)$$

Where,

$x_\emptyset = xcos\emptyset + ysin\emptyset$

$y_\emptyset = -xsin\emptyset + ycos\emptyset.$

From equation (7), $\emptyset$ is the orientation of the TGF, T is the period of the sinusoidal plane wave and σ is the standard deviation along the axis as mentioned earlier. Further decomposing (7) result in the formation of two orthogonal parts. That is, one parallel while the other will be perpendicular to the orientation $\emptyset$ and this would be represented as;

$$\text{Re } g(x,y,T,\emptyset) = h_x(x;T;\emptyset) \cdot h_y(y;\emptyset)$$

$$= \left\{\exp\left(-\frac{x_\emptyset^2}{2\sigma_y^2}\right)\cos\left(\frac{2\pi x_\emptyset}{T}\right)\right\} \cdot \left\{\exp\left(-\frac{y_\emptyset^2}{2\sigma_y^2}\right)\right\} \quad (8)$$

The design of a filter bank consists of selecting a proper set of values for the filter parameters: $\emptyset$, σ and γ.The possible combination of the various parameters determines how the filter bank analyzes the spatial domain [*4*].

The GF finds its application in many areas of interest, for example Zhou et al., discussed the application of GF in face recognition [*18*]. In their publication, the 2-dimension GF (equation 6) was expressed as follows;

$$O_{[\lambda,\theta,\varphi,\gamma,B]}(u,v) = I(u,v) * G_{[\lambda,\theta,\varphi,\gamma,B]}(u,v) \quad (9)$$

Where $O_{[\ ]}$ is the Gabor wavelet, while u and v are the coordinates of the Gabor wavelet matrix pixel. Variable λ is the wavelength, θ is the orientation, φ is the phase offset, γ is the aspect ratio and B is the bandwidth. The facial features were extracted by convolving the GF with the image area $I(u,v)$. Figure 2.6 shows the operations performed in the convolution process.

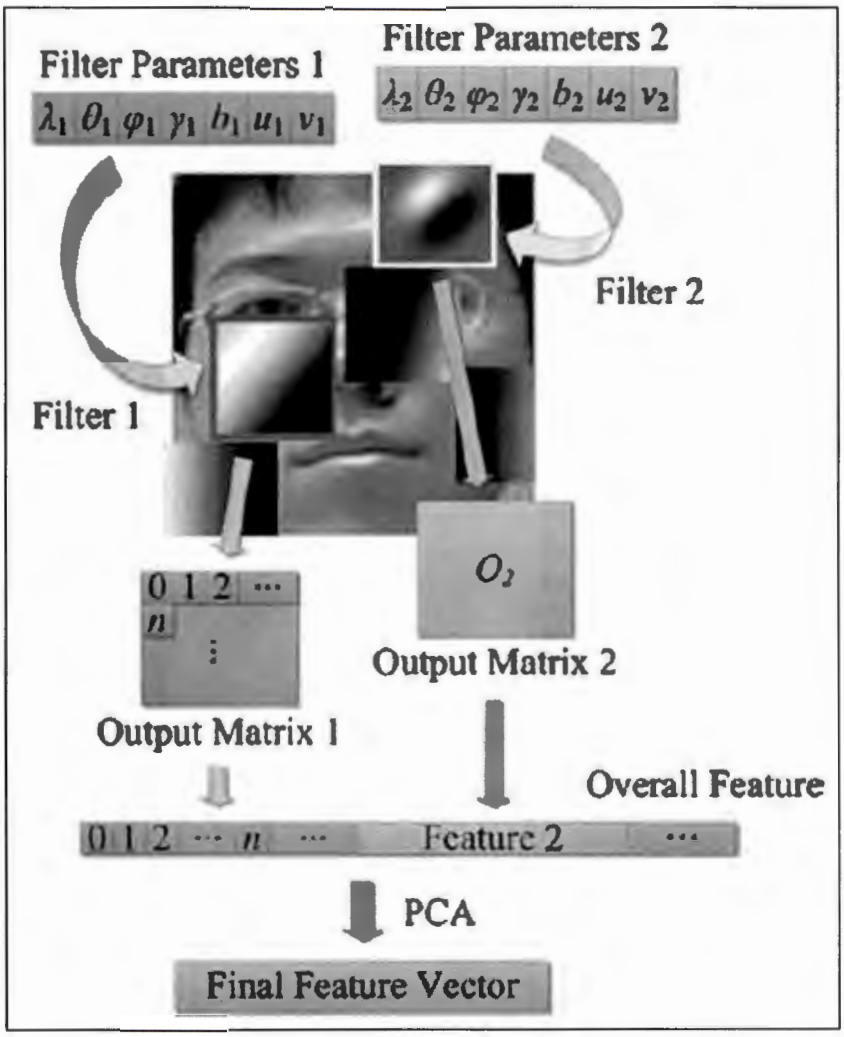


Figure 2.6: Gabor Filter Feature Extraction [*18*]

As shown in Figure 2.6, each convolution operation produces a feature matrix with its correlated feature data obtained by connecting the pixel values from the matrix into a 1-dimension vector. This means that if, for example a Gabor filter set with filter number of m and a wavelet matrix size of $m \times n$ pixels. The overall feature vector would be in the dimension of $m \times n^2$.

2.5.3 Modified Gabor Filter

The Modified Gabor Filter (MGF) was primarily inspired by the drawbacks of the TGF, chief among them being its image-dependent parameter selection strategy [*2*]. To overcome this

drawback, the MGF replaced the cosine function cos(x; T) in equation (8) with a periodic function $F(x; T_1; T_2)$, to help reduce loss of useful original information resulting from failure by the TGF to pass the entire frequency harmonic through the filter. The period T was extended to periods T_1 and T_2 representing the regions above and below the axis respectively as illustrated in Figure 2.7.

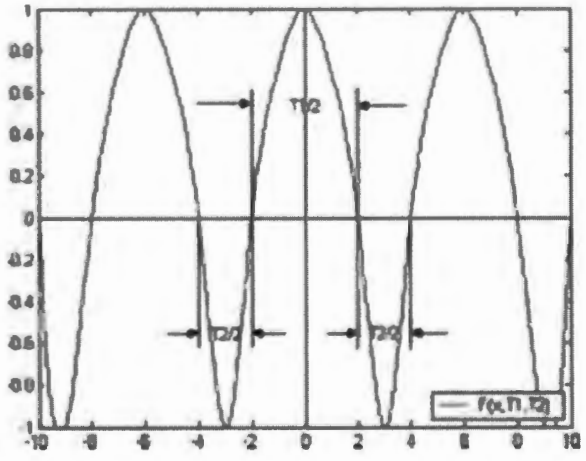


Figure 2.7: The periodic function $\mathbf{F(x; T_1; T_2)}$ with cosinusoidal functional curve with periods $\mathbf{T_1}$ and $\mathbf{T_2}$ [2].

Figure 2.7 show a periodic even-symmetric oscillator with the period $(T_1 + T_2) / 2$ which becomes a true cosinusoidal function when $T_1 = T_2$. Which could be expressed as function where $|t| = floor(t)$, representing the largest integer values of the selected feature not larger than t as follows:

$$f(x) = \begin{cases} cos(2\pi x/T_1) & , 0 \le x \le T_1/4 \\ -cos(2\pi\,(x - T_1/4 - T_2/4)/T_2) & , T_1/4 < T_1/4 + T_2/4 \\ cos(2\pi\,(x - T_1/4 - T_2/4)/T_2) & , T_1/4 + T_1/2 \le x \le T_1/2 + T_1/2 \end{cases} \quad (10)$$

Modulating the periodic function $F(x; T_1; T_2)$ by a Gaussian function resulted in equation (8) expanding as follows;

$$g'(x, y, T_1, T_2, \emptyset) = h'_x(x; T_1, T_2; \emptyset) \cdot h'_y(y; \emptyset)$$

$$= \left\{\exp\left(-\frac{x_\emptyset^2}{2\sigma_y^2}\right) F(x_\emptyset, T_1, T_2)\right\} \cdot \left\{\exp\left(-\frac{y_\emptyset^2}{2\sigma_y^2}\right)\right\} \quad (11)$$

The representation in equation (11) transformed the MGF into a band pass filter associated with a bank of low pass filters which allow retention of useful low frequency components that were ignored by the GF.

The implementation MGF followed its design that was completed based on its analysis in frequency domain while images are enhanced in spatial domain. The algorithm considered implementing image rotation as opposed to multi-directional MGF where its coefficients had an orientation of $\theta = 0$. That is, MGF banks with different $\sigma_x s$ corresponding to different periods T1 and T2 are computed first then the image blocks with the same size as that of the convolution mask are rotated to the MGF orientation at $\theta = 0$ [*19*]. This resulted in the MGF achieving better results as compared to the TGF as shown in Figure 2.8.

Figure 2.8 (a) and (b) illustrate results from the GF with parameter values $\sigma_x = 2.0$, $\sigma_y = 4.0$ and $\sigma_x = 2.5$, $\sigma_y = 4.0$ respectively, while Figure 2.8 (c) are the results obtained from the MGF algorithm.

Figure 2.8: Fingerprint Images Enhancement Result for the TGF and the MGF [*17*]

2.5.4 Mathematical Morphology

The Mathematical Morphology (MM) approach is based on set theoretical principles where the parameter of the morphological operations is a shape (set or function of any dimensions) commonly known as the Structuring Element (SE) [*20*].

The approach works by making a direct and figurative relationship between the original image to be processed and the SE using a combination of two basic operations, *erosion* and *dilation*. Each SE has a shape which can be thought of as a parameter to the operation and if the SE is appropriately selected with consideration to its size, shape and orientation then the operation is guaranteed of a better output image.

The morphological operators were initially developed to analyze binary images but have been elegantly extended to operate on gray-scale images, 3D images and graphs. This was made possible given that the binary and grey-scale pixels are scalar values that have a natural set of operations and ordering properties [*21*].

The two operations, erosion (reduction) and dilation (increase) are implemented using set theory which when applied to an image it result in either increasing or decreasing the intensity of the pixels. For example, dilation was defined as the maximum value in the window which improves the intensity of the resulting image by changing pixels with values *"0"* or *"1"* [*22*]. In that case, dilation of a gray-scale image $F(x, y)$ by a gray-scale SE $B(s, t)$ is denoted by;

$$(F \oplus B)(x, y) = max\{F(x - s, y - t) + B(s, t)\} \tag{12}$$

While on the other hand, erosion is seen as just the opposite of dilation (minimum value in the window), whereby the image after erosion will be darker than original image as the pixels with value of "1" will be turned to "0" and is denoted by;

$$(F \ominus B)(x, y) = min\{F(x + s, y + t) - B(s, t)\} \tag{13}$$

The MM approach was successfully applied in quite a number of areas in remote sensing images such as edge detection using multi-structure elements, meant to evaluate sets of brain and chest CT images, 2D and 3D image representation and vehicle identification among others [*22*] [*23*] [*24*].

Figure 2.9 shows a flowchart of a MatLab simulator as applied by Wei and Tong for edge detection [*25*].

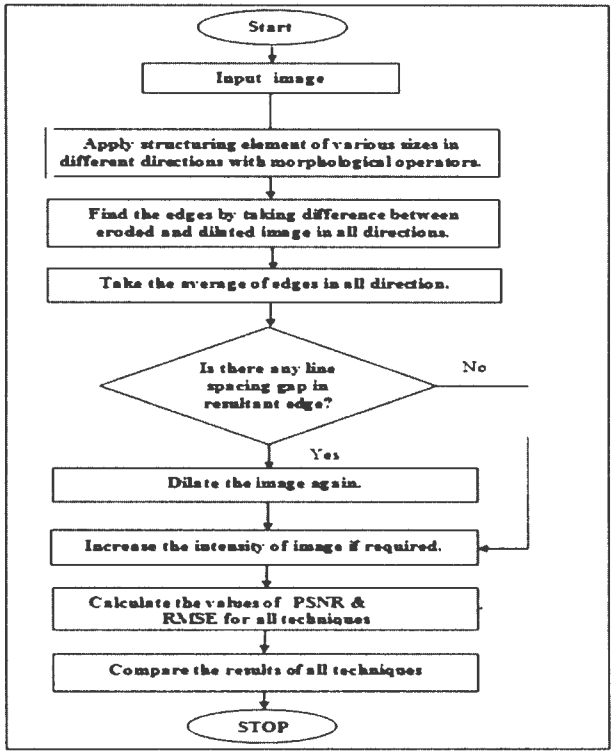


Figure 2.9: A MatLab flowchart for the Morphological Edge Detection Algorithm [*22*]

Opening and closing are as well morphological parameters that can be viewed as byproducts dilation and erosion operators. Opening is characterized by a combination of MM parameters in the following order, erosion followed by dilation of a given image. Closing can be viewed as the opposite of opening, that is, the image is initially dilated and later eroded. Like the erosion and dilation, the closing and opening operators as well implement a structuring element of any size and shape. Equations (13) and (14), respectively shows an implementation of the erosion and dilation operators over a binary image A by a given structuring element B.

$$(A \ominus B) = \{x | B_x^1 \subset X\} \tag{13}$$

$$(A \oplus B) = \{x | B_x^2 \subset X^c\} \tag{14}$$

In the erosion operator the hit and miss transformation establishes that a pixel x belongs to the eroded set if each point of the element B^1 translated to x be on X. This operator removes the pixels at the borders of the objects in the set X, thereby shrinking the resulting image. This operation finds its application mostly in removing noise from threshold images.

Figure 2.10 illustrates how the erosion operator works on a given original image applying a 3x3 structural element.

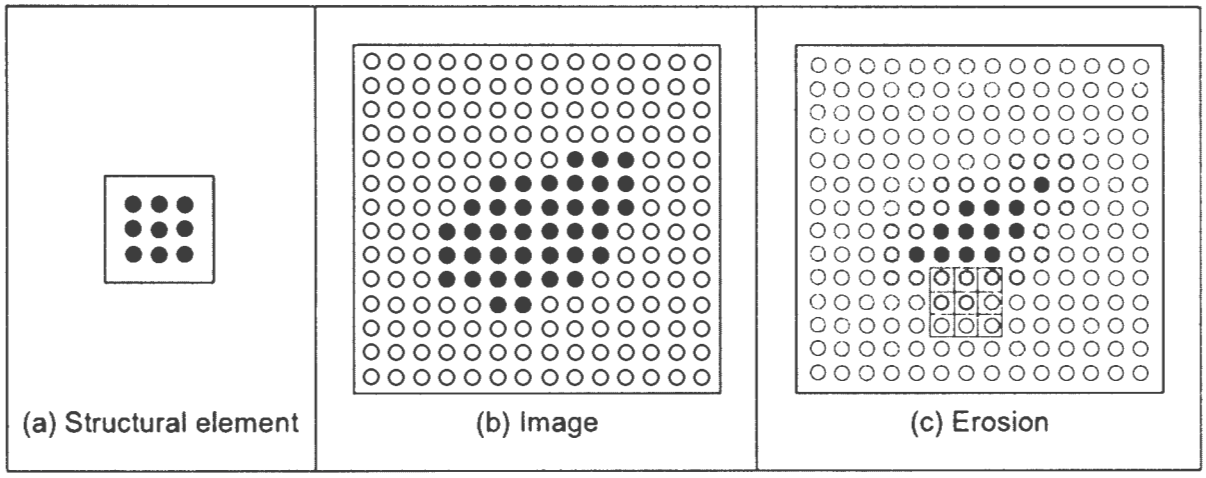


Figure 2.10: Example of the Erosion Operator [*26*].

The dilation operator defined in Equation (14) establishes that a point belongs to the dilated set when all the points in B^2 are in the complement. This operator erodes or shrinks the complement so that when the complement is eroded, the set X is dilated [***26***]. Figure 2.11 illustrates the dilation process with the structural element shown in Figure 16(a) defining the set B^2.

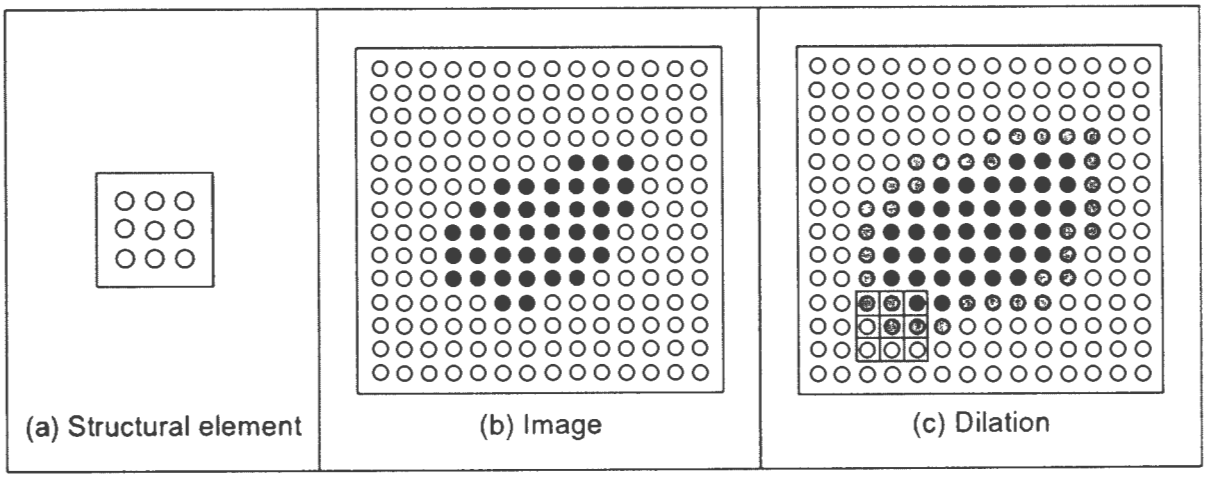


Figure 2.11: Example of the Dilation Process [*26*].

2.5.5 Principal Component Analysis

The Principal Component Analysis (PCA) can be viewed as a statistical procedure for interpreting the covariance structure of a set of observations of possibly correlated variables by identifying the principal directions in which the data in a given set varies [*27*]. The principal directions sometimes referred to as principal components are a result of transformation defined in such a way that the first principal component has the largest possible variance and each succeeding component in turn has the highest variance possible in that set. Figure 2.12 shows the PCA for data representation, where q represents the principal direction in which the data varies and p represents the axis orthogonal to q. x and y are the coordinate system of the Cartesian plane.

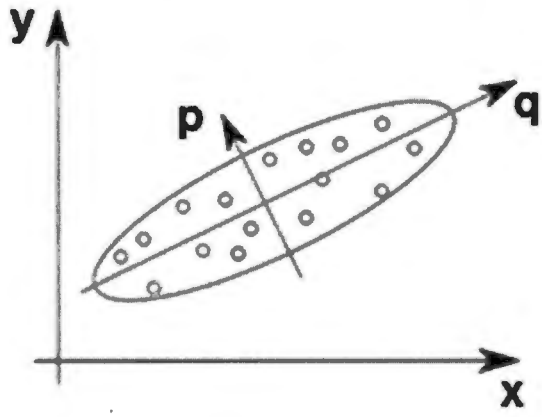


Figure 2.12: Data Representation in PCA

When using the PCA, any given image, say image A, can be viewed as a vector of pixel values which can be transformed using different mathematical transformations such as translation and scaling into subspaces that may serve as a feature extraction [*28*].

In computational terms, the PCA as discussed in [*28*], follows certain basic steps as below;

(i) *Select a set of training images, say A defined as a* $(n \times n)$ *matrix and compute the PCA basic steps.*

(ii) *Create data samples by subtracting image A from the training set as follows;* $a_1, a_2, a_3, \ldots, a_n \epsilon A - A^1$

(iii) *Calculate the covariance matrix between data samples as,* $cov = (A * A^1)/(N-1)$ *where N is the number of columns in the data sample (n).*

(iv) *Find the Eigen value and vector of the covariance matrix (cov) as the determinant of* $(cov - \lambda I)$*, that is* $|(cov - \lambda I)| = 0$*, where I is the identity matrix.*

(v) *Identify the diagonal Eigen elements and extract the largest Eigen values from the identified elements.*

2.5.6 Scale Invariant Feature Transform

Scale Invariant Feature Transform (SIFT) is an algorithm mainly used to detect and describe local features in images, mostly applied in object recognition, image sketching, 3D modeling, gesture recognition and video tracking [***11***]. It was introduced by David Lowe in [***29***]. The algorithm uses the image keys to identify possible object matches and to verify each match by finding a low residual least-squares solution for the unknown model parameters. These identified features are said to be invariant to geometric transformations such as scaling, translation and rotation.

Lowe in [***29***], discussed the SIFT as an approach that takes an image, transform it into a large collection of local feature vectors and then apply a four stage filtering approach, that is; *Scale-Space Extrema Detection, Key-point Localization, Orientation Assignment and Key-point Descriptor.*

2.5.6.1 Scale-Space Extrema Detection

The first stage in filtering which attempts to identify those locations and scales that is identifiable from different views of the same object. It makes use of "scale space" function that is based on the Gaussian function as defined in equation (16).

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{16}$$

Where * is the convolution operator, $G(x, y, \sigma)$is a Gaussian variable scale and $I(x, y)$ being an input image. Equation (16) identifies candidate key-points by finding the difference between the transformed image with scale k and the original image as follows;

$$D(x,y,\sigma) = \big(G(x,y,k\sigma) - G(x,y,\sigma) * I(x,y)\big)$$
$$= L(x,y,k\sigma) * L(x,y,\sigma) \quad (17)$$

Finally, extrema of $D(x,y,\sigma)$ are determined by comparing each point with its 8 plus or minus 1 (+/-1) neighbors at the same scale k.

2.5.6.2 Key-point Localization

This stage performs a detailed fit to the nearby data for location, scale and ration of principal curvatures [***29***], thereby rejecting points with low contrast or poorly localized along the edge. Equation (17) calculates the fitness of the candidate points that will be compared to the threshold value for acceptance or rejection.

$$z = -\frac{\partial^2 D^{-1}}{\partial x^2}\frac{\partial D}{\partial x} \quad (18)$$

2.5.6.3 Orientation Assignment

At this stage, the magnitude or gradient $m(x,y)$ and orientation $\theta(x,y)$ are calculated in order to use the histogram of gradients around the feature point for representation. The magnitude and orientation are calculated as shown in Equations (18) and (19) respectively.

$$m(x,y) = \sqrt{((L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \quad (19)$$

$$\theta(x,y) = tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y))) \quad (20)$$

Where L the Gaussian is smoothed image.

2.5.6.4 Key-point Descriptor

The previous three stages worked on assigning an image location, scale and orientation respectively to each given key-point. Therefore at this stage the idea was to compute a descriptor for the local image region that is highly distinctive to the remaining candidate points. Lowe and Santhseelan et al. in [***29***] and [***11***] respectively, discussed the use of gradient data computed in equations (19) and (20) to create key-

point descriptors. They used the data obtained from $m(x, y)$ and $\theta(x, y)$ to construct a set of histograms over a window centered at the key-point with a weight of 1.5 times the scale of the key-point added.

The peak of the orientation histogram was then used to select the best candidate points in the histogram. Only those that fall within the 80% range were considered as valid points which in turn, result in the creation of another set of candidate key-points to be considered. This would repeat until the best candidate point is obtained. Figure 2.13 shows a 2x2 descriptor array that was computed from an 8x8 set of samples using gradient and orientation data $m(x, y)$ and $\theta(x, y)$ respectively.

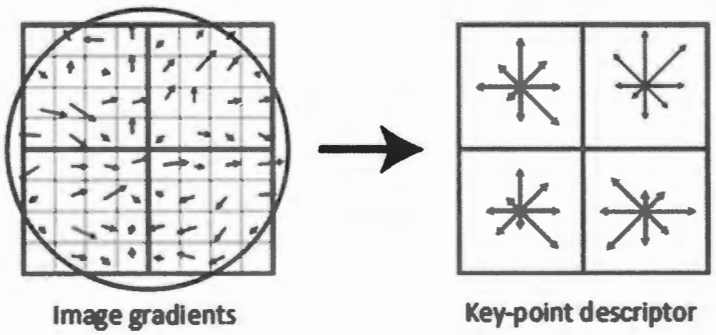


Figure 2.13: Computation of the key-point descriptor [*29*]

2.5.7 Hough Transform

The Hough Transform is a mathematical transformation of points from the input space, referred to as the feature space into curves in parameter space that can be used for the detection of geometric patterns [***30***]. This method is based on the fact that all points from a straight line that are positioned in feature space can be mapped to a single point in parameter space.

The Hough Transform has three variances that can be considered depending on the nature of the problem being analyzed. These are the Generalized, the Randomized and the Hybrid Hough Transform. This technique uses an N-dimensional array called *"the voting array"* for the event, where N is the dimensionality of the hypothesis space. The voting array is used to

capture the normalized conditional probabilities so that each event simply adds its own voting array into the accumulator representing the hypothesis space [*30*].

The following steps represent the generalized Hough transform algorithm;

1. *Create a fixed grid representing the parameters* (x, y) *that need to be estimated.*
2. *At each point on the grid, the likelihood of the estimate being* (x, y) *given the measurement* $\beta, p(x, y|\beta)$*, is evaluated and accumulated in an array, A:*

$$A(x, y) = \frac{1}{L}\sum_{l=1}^{L} p(x, y|\beta_i) \quad (21)$$

Where β_i *is the* l^{th} *measurement of the total L.*

3. *Consider the estimate as the grid position corresponding to the peak accumulated likelihood.*

The Generalized Hough Transform technique find its application in different domains such as elimination of ghost detections in target tracking using multi-sensor data diffusion, self-positioning of robots using map matching and emitter geo-location using time difference of arrival [*30*], [*31*], [*32*]. Its application has in most cases produced acceptable results as elaborated in section 5.1.5 of this thesis.

2.5.8 The Linear Discriminant Analysis

The Linear Discriminant Analysis (LDA) sometimes known as the Fisher's Linear Discriminant Analysis as discussed by Pali et al. is an appearance based approach feature extraction technique [*14*].

The LDA technique works by finding a set of projecting vectors *w* that best discriminate different classes. According to the Fisher criteria as presented by Pali et al., the LDA can achieve this by maximizing the ratio of determinant of the between-class scatter matrix S_b and the determinant of the within-class scatter matrix S_w. The within-class scatter S_w is a representation of how the face images are distributed closely within-classes while the between-class scatter matrix S_b represents how the classes are separated from each other.

The idea behind the LDA technique is to divide the training images into different classes that is, the S_b and the S_w. This followed by finding a set of vectors that maximizes the fisher's discriminant criterion and this is achieved by maximizing the S_b on one hand while minimizing S_w on the other.

When applying the LDA for example, we may consider two sets of points Figure 2.14, green (points in rectangular form) and blue (points in circular form) in color that are presented in two-dimensional space projected onto a single line. If we are to rely on the direction of the line, the points can either be combined together as shown in Figure 2.15 or be separated as shown in figure 2.16. In this case the LDA tries to find the line that best separates these sets of points. Figures 2.15 and 2.16 illustrate some examples of poor and good separation of points respectively as scenarios that can be presented by the LDA.

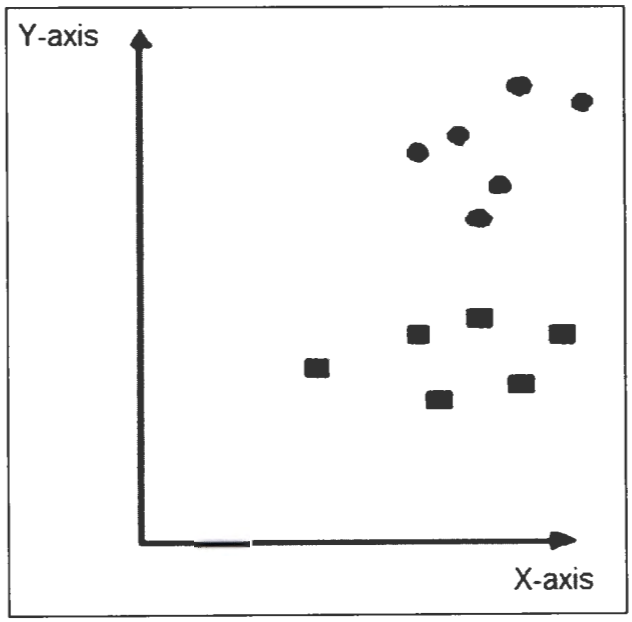


Figure 2.14: Points in 2D Space [*14*].

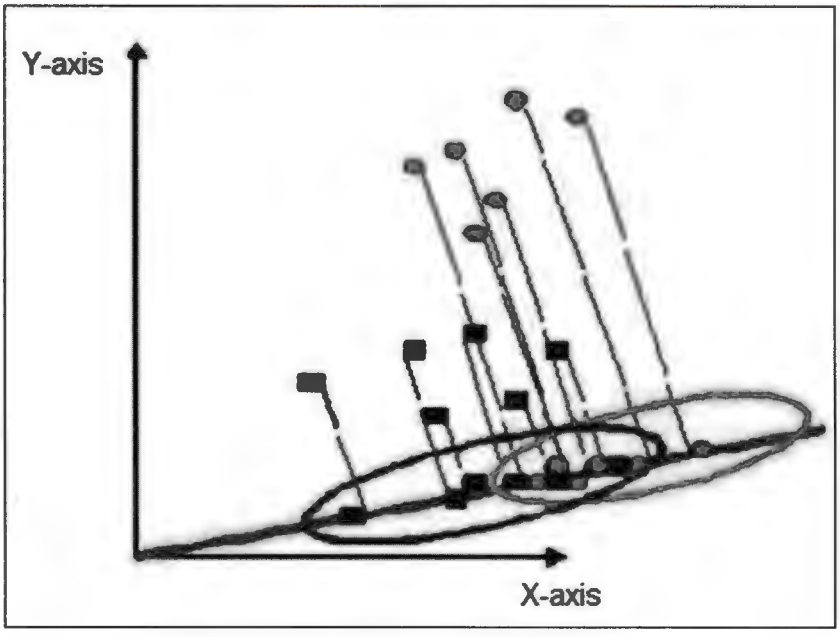


Figure 2.15: Example of a Poor Separation [*14*].

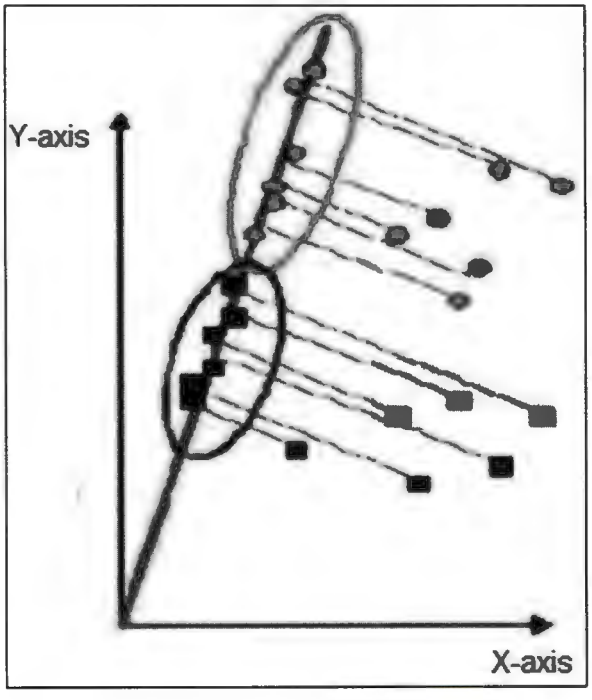


Figure 2.16: Example of a Good Separation [*14*].

Applying the LDA in face recognition domain the points represents facial images and the within-class (S_w) scatter matrix, also known as intrapersonal matrix, illustrates variations in the appearance of the same individual due to different lighting illuminations and facial expression, while the between-class (S_b) scatter matrix, also known as the extra-personal,

depicts the changes in appearance due to a difference in the identity. In this way fisher-faces can project away some variation in lighting and facial expression while maintaining discriminability [*33*]. The LDA technique suffered some challenges with small sample sizes in which the with-in class scatter matrix becomes singular and thus the technique fails to interpret. According to Pali et al. a number of a number of extended LDA algorithms were suggested and among them the most popular one is the PCA (discussed in 2.5.5) for dimension reduction prior to performing LDA.

2.6 Parameter Selection Strategies

Parameter selection plays a crucial role in the use of FET, as the choice of "good" parameters gives a "good" output image. This section brings an overview of the parameter selection strategies employed by various FET.

2.6.1 The Traditional Gabor Filter

A strategy that selects filter parameters semi-automatically using the Information Diagrams (ID) concept was explored in [*10*]. The strategy considered only the orientation ($\emptyset$), aspect ratio (γ) and sigma of the Gaussian envelope (σ) while the rest of the parameters remained fixed and were specified based on empirical data [*19*]. The three parameters were selected by looking for a local extrema. For example, for each $\emptyset$ in the set we compute the parameters of the highest local maximum and smallest local minimum as follows:

Highest Local Maximum:

$$(\sigma_i^{max}, \gamma_i^{max}) = \arg\max_{\sigma,\gamma} \emptyset - \ ID_{x,y}^{\emptyset_i} \tag{22}$$

Smallest Local Minimum:

$$(\sigma_i^{min}, \gamma_i^{min}) = \arg\min_{\sigma,\gamma} \emptyset - \ ID_{x,y}^{\emptyset_i} \tag{23}$$

From 3 and 4, the set of chosen GF parameter can be expressed as follows:

$$P_{\emptyset}^{min} = \{(\sigma_i^{min}, \gamma_i^{min}, \emptyset_i), \ldots, (\sigma_n^{min}, \gamma_n^{min}, \emptyset_n)\} \quad (24)$$

and

$$P_{\emptyset}^{max} = \{(\sigma_i^{max}, \gamma_i^{max}, \emptyset_i), \ldots, (\sigma_n^{max}, \gamma_n^{max}, \emptyset_n)\} \quad (25)$$

Consequently, the selected feature vector representation becomes:

$$v_{(x,y)} = (v_{(x,y)}^1, \ldots, v_{(x,y)}^i 0, \ldots, v_{(x,y)}^m)^T \quad (26)$$

$$\text{Where, } v_{(x,y)}^i = \left|G_{\emptyset_i,\lambda_i,\sigma_i}(x,y)\right|, (\lambda_i, \emptyset_i \sigma_i) \in P_{\emptyset}^{max} \cup P_{\emptyset}^{min} \quad (27)$$

Table 2.1 shows parameter values for $\emptyset, \sigma$ and γ computed for each ID, the distance metrics (Euclidean & Mahalanobis) and the feature model type (magnitude versus real-imaginary parts).

Test ID	type	# local max	# local min	distance	mag	re+im
1	θ	1	1	Mah	68.49	78.33
2	θ	2	0	Mah	85.92	95.83
3	γ	2	0	Mah	58.19	74.16
4	γ	1	1	Mah	54.41	75.83
5	σ	2	0	Mah	58.19	72.50
6	σ	1	1	Mah	50.21	72.50
7	θ	1	1	Euc	31.93	85
8	θ	2	0	Euc	38.87	87.5
9	γ	2	0	Euc	17.86	53.33
10	γ	1	1	Euc	15.55	45
11	σ	2	0	Euc	24.79	74.17
12	σ	1	1	Euc	15.97	75.83

Table 2.1: Computed values for parameters $\emptyset, \boldsymbol{\gamma}, \boldsymbol{\sigma}$ using Information Datagrams [*10*]

2.6.2 The Modified Gabor Filter

Yang et al. [*19*] in their publication used an approach whereby the period T of the GF was decomposed into T_1 and T_2 and then specified the remaining parameters $\emptyset$, σ_y and the convolution mask size adaptively.

2.6.2.1 Specifying Orientation $\emptyset$

The original image is divided into blocks of sizes W x W and the parameter $\emptyset$ is specified as the orientation of each pixel in the block using the following formula;

$$\emptyset(i,j) = \frac{1}{2}\tan^{-1}\left(\frac{\sum_{u=i-w/2}^{i+w/2}\sum_{v=j-w/2}^{j+w/2} 2G_x(u,v)G_y(u,v)}{\sum_{u=i-w/2}^{i+w/2}\sum_{v=j-w/2}^{j+w/2}\left(G_x^2(u,v)-G_y^2(u,v)\right)}\right) \quad (28)$$

In Equation (28), W is the size of each subdivided block, G_x and G_y is the local gradient at each pixel in each subdivided block. The resulting orientation value of $\emptyset(i,j)$ obtained from Equation (28) is later regularized into the range $\left[-\frac{\pi}{2}, +\frac{\pi}{2}\right]$.

2.6.2.2 Specifying Standard Deviations σ_x and σ_y

The two parameters σ_x and σ_y were specified differently owing to their effects on the output image. The parameter σ_y was empirically set to 4.0 and σ_x was of great concern as its performance is related to the periods T_1 & T_2 thereby influencing the degree of contrast on an image [*19*]. As a result, the periodic function represented by $h_x(x; T_1, T_2; \emptyset)$ Equation (11) had to be examined for constraints on regions below the x-axis and those close to the origin above the x-axis.

The values of T_1 & T_2 were computed as in 2.3. Period T_2 was subdivided into a smaller range T_1. In Table 2.2 the circled column shows some of the σ_x results obtained from the experiments.

T_1	T_2	σ_x
4	[4, 12]	1.5
4	[14, 18]	1.6
4	[20, 28]	1.8
6		1.8
8	--	2.5
10	--	2.7
12	--	3.0
14	--	3.5
16		4.0

Table 2.2: Computed σ_x values for different T_1&T_2periods [*19*].

2.6.2.3 Specifying periods $\boldsymbol{T_1}$ *and* $\boldsymbol{T_2}$

By analyzing Figure 2.17, we can infer that function $F(x;T_1;T_2)$ is a periodic even-symmetric oscillator with the period $(T_1 + T_2)/2$, given that from the figure $\frac{T_1}{2}$ and $\frac{T_2}{2}$ corresponds to the regions above and below the x-axis respectively.

The periods T_1 and T_2 were specified by initially identifying a pixel that needs to be filtered, say, $P_a(x,y)$ and noting its colour intensity. This was followed by continuously checking the neighboring pixels until you find a pixel with different colour intensity and denote it $P_b(x_1,y_1)$. T_1 was set to $2W_1$ the distance between $P_a(x,y)$ and $P_b(x_1,y_1)$ while T_2 was set to $2W_2$, that is, the distance between $P_b(x_1,y_1)$ and the next pixel $P_c(x_2,y_2)$ with a colour intensity different from $P_b(x_1,y_1)$. Figure 2.17 shows how the periods T_1 and T_2 were calculated on a fingerprint image obtained from an image database [*19*].

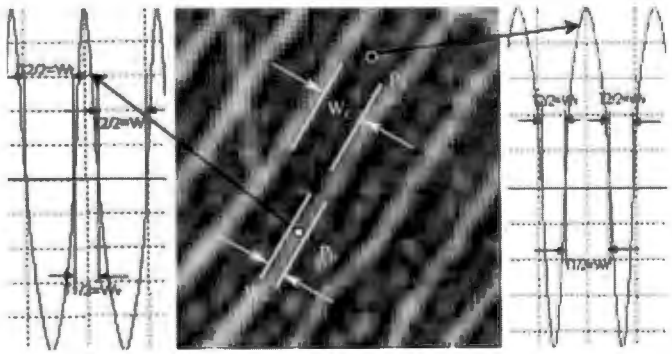


Figure 2.17: The Curve of $F(X;T_1;T_2)$ Corresponding To Different Periods T_1 and T_2 [*19*]

2.6.3 The Mathematical Morphology

Given that the parameter of the morphological operations is a set or a function of any dimensions known as the SE, Fejes and Vaida in [*20*] used an adaptive approach to define the SE and its regions. The approach applied a technique that was based on the method of the Least Mean Square (LMS) which minimizes the error calculated between the output and a selected desired signal (Mean Square Error).

According to the LMS coefficients-update, the following equation was used on simple grey scale dilation.

$$b'_m = b_m + 2\mu(d - y) \text{ and } b'_i = b_i \quad (29)$$

Where $i, m \in B$ and $m \neq i$ B represent the support of the SE. b_i and b_m are parameters to be optimized, μ is the convergence parameter, while y and d denote the output and the reference signal respectively [*20*].

On the other hand a statistical analysis of the SE templates B_i is used to define the regions of interest as follows;

$$B = \{B_i\} \quad (30)$$

This is followed by finding tuples (R_k) of parameter locations in an arbitrary B_i template so that a value B_{pi} of B_i at location p does not differ from another element of the same tuple more than an error limit of λ for all i as indicated in (31);

$$\forall p, q \in R_k^{\lambda}, \text{ iff } \forall i \; \delta_{pqi} < \lambda, \text{ where } \delta_{pqi} = \left(b_{pi} - b_{qi}\right)^2 \quad (31)$$

When implementing adaptation, the coefficient-update for parameter regions are determined by identifying an element m such that,

$$\text{if } m \in R_k^{\lambda} \text{ then } r'_k = r_k + 2\mu(d - y) \quad (32)$$

where r_k and r'_k denote the old and the new value of the k^{th} parameter region respectively and r'_k can be any single parameter located in the region being extracted.

2.6.4 The Scale Invariant Feature Transform

The use of image keys is one fundamental concept when using the SIFT technique to identify candidate object matches. However, the reliability of establishing the set image correspondences plays an important role in accuracy of the estimated parameters [*34*].

Othman et al., proposed the use of a Detection Algorithm (DA) as discussed in 2.6.4.1, while Guang-hui et al., proposed the use of a Self-Adaptive SIFT Algorithm (SASA) as discussed in section 2.6.4.2 to estimate parameters for the SIFT technique [*35*] [*36*].

2.6.4.1 Detection Algorithm.

The DA algorithm is implemented following the steps below;

(i) *Find an initial position* (x_s, y_s) *called "the seed" inside the RoI and creates a rectangular boundary R around the seed.*

(ii) *Trace a* 10×10 *mask over the given image to determine the seed point* (x_s, y_s).

(iii) *Calculate the sum, standard deviation of each mask, it's correlation with its adjacent masks (neighbors) and consider one with minimum values as the mask containing seed point.*

(iv) *Consider the seed point as the center and construct rectangles* $(n \times n)$ *around the seed point* (x_s, y_s) *incrementing their size step by step, (ie.*10×10*).*

(v) *Repeat (iv) until the current standard deviation of one mask is greater than the standard deviation of the previous mask.*

Although the algorithm worked well in situations where it was implemented, however its short coming is that it relies mostly on empirical knowledge [*35*].

2.6.4.2 Self-Adaptive SIFT Algorithm.

The SASA algorithm as discussed by Guang-hui et al. in [*36*], focused on calculating and determining the thresholds automatically based on the image information, as well as changing parameters for the SIFT algorithm so that it increases the number of extraction features and matching points. This is achieved using the following steps (i) – (iii).

(i) *Calculate the average grey-scale* (AG) *value of input images and then divide the result by 10, that is* $AG = avgI(x, y) / 10.$ (33)

(ii) *Check the following, if* $(AG < 1.0)$ *reduce the contrast threshold and increase the number of features points to match.*

(iii) *Increase the curvature ratio of the threshold if the matching points are less than 8.*

2.6.5 The Principal Component Analysis

Li et al., proposed a method that estimate parameters based on a quantitative measure of practical identification of features [*37*]. In their method, the first principal component is the Eigen-vector that holds the highest Eigen-value in the matrix representation. The remaining candidate parameters are ranked using a recursive algorithm based on how they are identified as follows;

(i) *Select the highest ranked parameter* p_1 *from the candidate parameters for* θ_j*, such that,* $p_1 = \{\theta_k | E_j = max_j E_j\}$ *and set the number of selected parameters to* $n = 1$*, where* E_j *is the overall effect of each parameter on* θ_j*.*

(ii) *Compute the linear-dependence metric* d_j *for each remaining parameter* θ_j *with respect to previously selected parameters* $\{p_1, \ldots p_n\}$ *for all selected candidate parameters.*

(iii) *Calculate the identifiability index* I_j *for each remaining parameter as* θ_j*;*

$$I_j = E_j d_j \quad (34)$$

(iv) *Select the next highest ranked parameter* p_{n+1} *and repeat from step (i) through to (iii), otherwise stop iteration when all candidate parameters are selected.*

2.6.6 The Hough Transform

The Hough Transform is a 2D linear non-coherent operator which maps an image to a parameter domain in such a way that when the aim of the analysis is to detect straight lines in

an image, the parameter of interest completely defines the straight lines [*38*]. The technique uses the following equation to map the point (x, y) into parameters (r, t) which represent a straight line passing through the point (x, y).

$$r = xcos(t) + ysin(t) \tag{35}$$

When applying this technique, each pixel in the original image is transformed in a sinusoid in the (r, t) domain and the presence of a line is detected by the location in the (r, t) plane in which more sinusoids intersect. This leads to all the straight lines in the original image being represented by a peak in the Hough domain. In most applications, the Hough Transform uses the Mahalanobis distance to calculate the nearest distance between the subject's image and its corresponding template image for feature extraction. This distance depends on the correlation between the covariance matrixes of the two input images [*39*]. The Mahalanobis Distance is given by equation (36) as follows;

$$D_{Mahalanobis}(x, y) = \sqrt{((x - y)^T \sum^{-1}(x - y))} \tag{36}$$

Where,

D : Distance between two images,
x : Subject's image,
y : Corresponding template image,
$\sum^{-1}$: Covariance matrix.

2.7 Application of GA in Remote Sensing Images Processing

Genetic algorithms were pioneered by Holland and have been proven to be very useful in a variety of search and optimization problems over the years [*40*]. Du et al in [*41*], discussed genetic algorithm as a search technique used in computing to find approximate solutions to optimization and search problems [*41*]. The evolution process in most cases starts from a randomly generated population of individuals that are iterated over several controlled generations.

In each generation, the fitness of every individual in the population is evaluated and based on their fitness values, multiple individuals are stochastically selected from the current population. These individuals are then modified by recombining and sometimes mutating them to form a new population which will be used in the next iteration of the algorithm. The iterations in most cases terminate either when the maximum number of generations has been reached or a satisfactory solution or fitness level has been obtained.

Genetic algorithms finds their applications in a number of areas in the extraction of unstructured data [*41*], extraction of global motion estimation data [*42*], extraction pathological voice data for classification [*43*] or implemented as filters for mobile robot localization and mapping [*44*].

The use of genetic algorithms proved to be a success given that in most of literature we reviewed, it was evident that the use of genetic algorithm improved performances in al algorithms they were implemented. Most of the authors fused genetic algorithms to previous versions of the algorithms to form a combinational approach that achieved by far better performances as compared to other algorithms. This was evidenced by results obtained in their simulations and/ or experiments.

In [*41*], genetic algorithm were modified to extract features from an unstructured data in an efficient and convenient manner. The modified genetic algorithm was tested using a set of unstructured data composed of words from a dictionary of which it proved to be efficient and effective especially in precision and recall using some formulae as shown in Table 2.3.

Test text	words	The original dimensions after word segmentation	The dimensions after genetic algorithm	Precision	Recall
Text 1	325	107	41	48.6%	84.5%
Text 2	563	156	73	45.4%	83.1%
Text 3	781	282	126	43.3%	85.2%
Text 4	1033	324	153	40.6%	85.6%
Text 5	1965	623	304	31.7%	88.9%
Text 6	2507	817	398	27.1%	90.2%

Table 2.3: Test Results of Feature Extraction by the Modified Genetic Algorithm [*41*].

Rao et al., discussed the classical approach in global motion estimation where multiple real-world points of an image were extracted based on structural criteria such as edge and corners

and their novel approach which uses genetic algorithms to combine feature extraction and selection [*42*]. The GA proved to be effective in selecting more reliable features for tracking global motion and feature tracking ranging from generation of interest point operators to actual searching for features within a frame [*45*]. Figure 2.18 shows the performances of different approaches in extracting features from a road image. The dotted straight lines indicate the mean values of each approach's performances on each extracted frame.

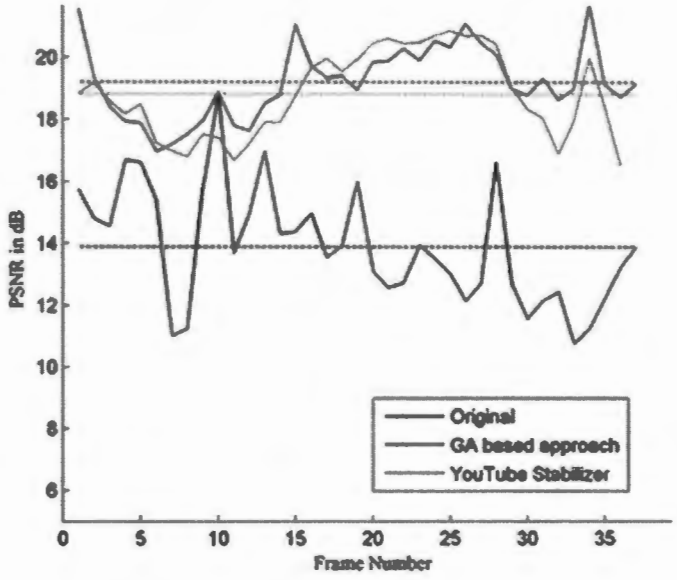


Figure 2.18: Comparative Stabilization Performance using PSNR [*42*].

Feng et al., presented an approach that implements genetic algorithms to solve the Simultaneous Localization and Mapping (SLAM) problem in mobile robots by concurrently optimizing appropriate cost functions defined over two sets of chromosome populations representing the robot pose and the environmental map. Simulations and experiments were conducted to demonstrate the performance gains achieved by their approach and its potential to yield globally consistent SLAM results [*44*].

While Hosseini et al., introduced an algorithm that implements genetic algorithms to discriminate voice pathologies signals from each other via adaptive growth of wavelet packet tree based on the criterion of Local Discriminant Bases (LDB). Genetic algorithm was mostly employed for the selection of the best feature set and Support Vector Machines (SVM) as a classifier to guarantee better results as much as possible. The experimental results show much

improvement in performance of their proposed approach and credit was given to the use of the genetic algorithm as combinational approach [*43*] .

Fuyan et al, discussed how they implemented genetic algorithms to improve the K-means algorithm. Their approach was to fuse the incremental genetic algorithms with K-means clustering. In this case genetic algorithms played an important role in simulating the evolutionary process of nature where solutions were evolved from one generation to the other thereby improving the final result. Their implementation followed some predefined iterated steps as illustrated in Figure 2.19 [*46*].

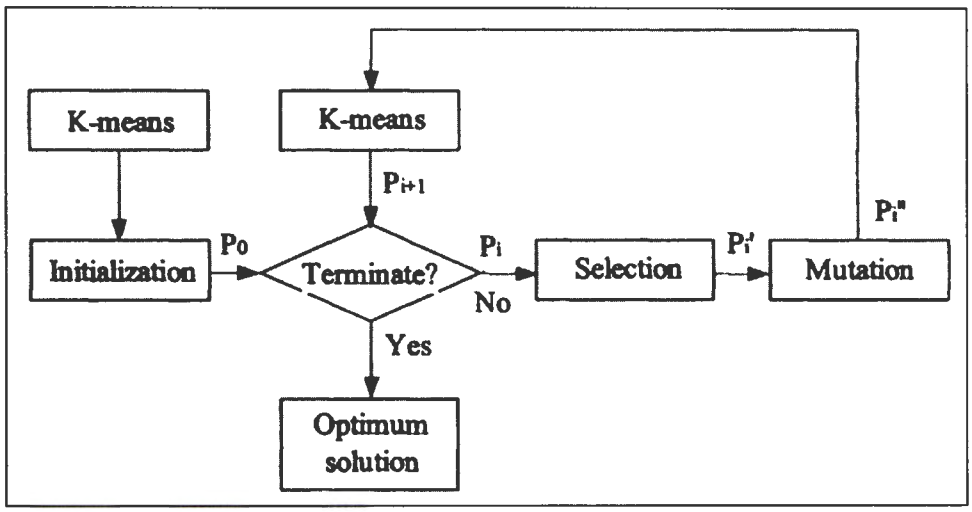


Figure 2.19: Genetic Algorithm Improvement Flowchart

It is quite evident from the literature we reviewed that genetic algorithms can be applied in feature extraction as an approach to improve the existing algorithms' performance. It is well important to mention that many traditional optimization techniques such as gradient based, simplex based methods and simulated annealing are mono-objective, hence they are not flexible enough to be extended to multi-objective cases as they are not conceived of to find multiple solutions to a given problem [*47*]. The use of genetic algorithms has helped in many cases where multiple solutions were required for a given problem. This notion gave us much needed confidence and assurance that our adaptive approach to parameterization of feature extraction techniques would succeed without much difficulties.

2.8 Chapter Summary

The chapter reviewed literature on the algorithms implemented by different FET, their parameter selection strategies and performance. The performance analysis we did was based on the simulations and experiments performed and discussed by different authors in their respective publications. We established that the TGF and the MGF empirically specified some of their parameter values depending on the output images.

However, the MGF tried to improve the process by splitting the period into regions, that is *T1* and *T2* then determine the RoI as discussed in section 2.6.2.3. Colour intensity played a major role in the determination of the periods. This to some extent significantly improved the results as compared to the TGF. The MM technique based much on the size of the SE, meaning to say the more appropriate the size of the SE the better the output results. While the PCA based its output image on the calculations of the parameter values using eigen-vectors and values.

The second section of the chapter saw us reviewing literature on the application of genetic algorithms in RSI processing with particular focus on feature extraction techniques. The literature we reviewed showed some successes in fusing GA with the current techniques, for example Table 3 showed an improved recall rate to as high as 90.2 %. This gave us some confidence in pursuing our research.

Chapter 3: Dynamic Parameterization of Feature Extraction Techniques

3.0 Chapter Overview

In this chapter we reviewed literature on approaches used to analyze and elicit requirements for dynamic systems. In this regard, five techniques were reviewed which are; the goal based, scenario based, use-case based, case based and ontology based. The review was meant to study and recommend an approach that could be used in the development of the GenApp. A case study was applied and a hybrid model was found and recommended for eliciting requirements for the development of the GenApp.

3.1 Preamble

Requirements Elicitation (RE) and specification are critical steps in every software requirements engineering process; hence a solid identification of stakeholders' needs and relevant documentation is crucial for the success of any project. A complete understanding of the stakeholders' problem domain, their clear and precise definition as well as a representation of the material that is understandable for all stakeholders helps to design software that really meet customers' needs [*48*]. A requirement by definition is an objective that must be met, while a specification describes how the objective is going to be accomplished. Analogically, a specification document describes how specific tasks are supposed to be carried out [*49*].

In most situations, stakeholders are non-technical experts and are therefore not quite familiar with the common techniques used in requirements engineering which are fairly abstract and often for them hard to grasp [*50*]. This is however in contrast with the developers who need a precise specification of the system structure and behavior for them to be able to analyze the requirements, validate stakeholders' needs and to formalize requirements for later verification procedures [*48*].

Cheng and Atlee, discussed elicitation and early modeling as collaborative activities that require the construction of a shared mental model of the problem and requirements [*51*]. This

has been applied in disciplines such as cognitive and social sciences where theoretical grounding and practical techniques are provided towards shared mental model and feasible graphical representation respectively, that are easily understandable by the stakeholders.

We found quite a number of RE techniques from the literature we reviewed that can be applied in algorithm development. However, in our study we confined to only the techniques discussed below. These techniques were considered based on their applicability to dynamic systems which characterizes our novel algorithm, the GenApp.

3.2 Goal-Based Requirements Analysis

Antón [*52*], viewed traditional systems analysis as ones that focus mainly on *what* features, such as activities and entities a system will support, whereas goal-based approaches focus on *why* systems are constructed, thereby providing the motivation and rationale to justify software requirements.

Goal-based approaches incorporate *goal analysis* and *goal evolution* as a way to address issues on how to identify goals in a system and what actions to take on the requirements when the goals change respectively. In a practical situation, goal analysis could be considered as a process of exploring gathered documentation ranging from information about the organization (enterprise goals) to system specific information (requirements) for the purpose of identifying, organizing and classifying goals [*52*].

In a software development process it is important especially from a developers' point of view that goals and requirements remain stable until the end of the project to avoid a lot of iterations in the refinement process. However, from experience this is only possible with goals which have a tendency of evolving at a gradual pace as opposed to requirements, which are a result of processes, organizational structures and operations that remain volatile in every organization. The gradual change of goals is usually motivated by elaboration and refinement processes whereby initially unforeseen constraints are considered or when synonymous goals are reconciled and/or merged. Such concerns are referred to as goal evolution.

3.3 Scenario-Based Requirements Analysis

Sutcliffe [*53*], defined scenarios as facts describing an existing system and its environment, that includes the behavior of agents and sufficient context information to allow discovery and validation of system requirements. As a result, scenarios contain different types of information, such as the operations of the current system and its environment, description of activities in case of manual systems and instance information as they describe real world events that happened to individuals.

Scenarios play important roles to both the developers and stakeholders during software development process as they help them understand and achieve some of their tasks and goals. In most situations scenarios can be used to guide an individual's thought or an idea as well as their reasoning.

As for stakeholders, scenarios can be used as test data to validate design models, while developers use them as a starting point for all the modeling and design that contributes to the design process and to test the models and their specifications during requirements validations process. Figure 3.1 illustrates a summarized view of some of the roles played by scenarios at each development stage.

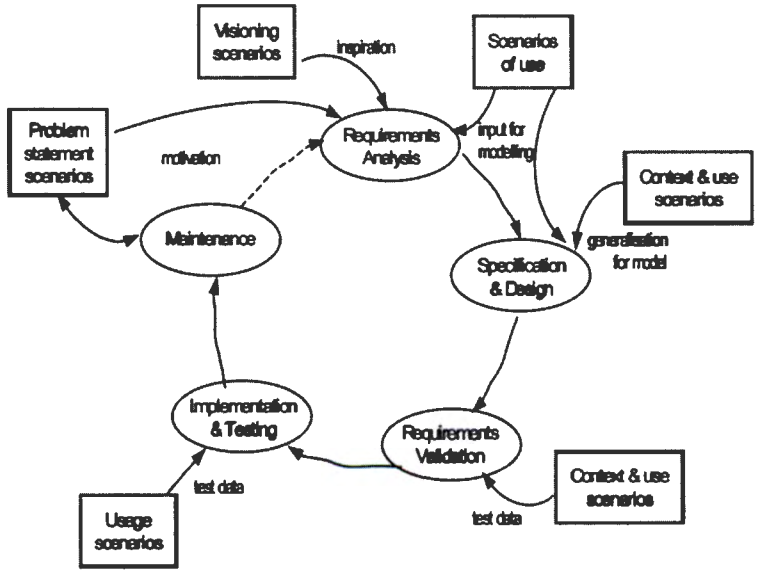


Figure 3.1: Scenario Roles in Requirements Specification [*53*].

3.4 Use Case-Based Requirements Analysis

The concept of Use Case (UC) was first proposed in [***54***] as a model used to define the interaction between external actors and subject system under consideration to accomplish a goal. The UC model is specified using a Unified Modeling Language (UML), as specified in figure 3.2, in a way that its diagrams drive the whole development process with the specification of client-valued functions that enable documentation of required system behavior in contractual agreements [***55***]. Ebnenasir et al., further defined UC by applying assertions as follows [***56***];

$$UC = \langle pre, post \rangle \Leftrightarrow pre = True \tag{37}$$

whereby, pre and $post$ are the preconditions and post conditions of the UC respectively. These conditions are the Boolean expressions which are specified in terms of subject system variables that are used to collect input and output values.

Considering Jacobson's proposal, we found out that the UC maybe triggered by *internal*, *external* or *temporal* events of which such triggers, in most cases give rise to systems errors [***54***]. These errors, if they remain unnoticed result in system bugs at a later stage hence the strong need to further explore the safety requirements when dealing with UC. In a way to reduce this challenge at design stage, *Misuse cases* (MUC) were introduced, with a sole purpose of specifying scenarios that the subject system must never exhibit. These MUC were represented as actions or events that negatively affect the system being modeled, such as system *crush* or *cyclic redundancy errors*. Formula 38 attempts to single out MUC in form of conflicting events or requirements thereby helping eliminate such anomalies at an early stage in system development. Equation (37) was further expanded into Equation (38), which allowed us to specify MUC as assertions in the same way as provided in Equation (37).

$$UC = \langle pre, post \rangle \;\; permits \;\; UC' = \langle pre', post' \rangle \Leftrightarrow (post \wedge pre') \not\equiv false \tag{38}$$

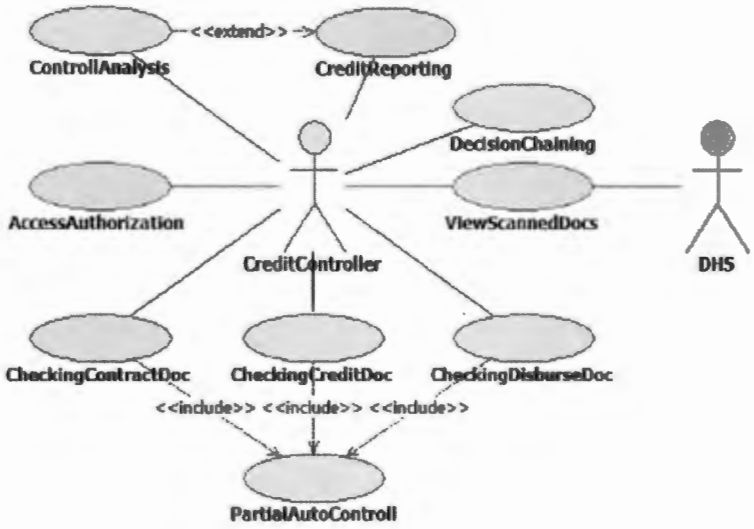


Figure 3.2: Example UC Diagram for Credit Controller Module [*55*].

3.5 Case-Based Reasoning Requirements Analysis

Jani et al., discussed Case-Based Reasoning (CBR) as a technique that reasons by remembering and recalling previously experienced cases in a similar way human beings reason when they are faced with new problems [*57*]. This technique uses a case base (CB), which is a repository of all past problems and their solutions. This is meant to speed up the Software Requirement Specification (SRS) process by way of reusing the knowledge and experience previously acquired and stored in the CB. The solutions retrieved from the CB can be used as they are (reuse) or with minor modification (revise) depending on the problem at hand. This approach serves processing time especially in situations where problems being solved are of similar domain.

The CBR functions by following a cycle of four steps, which are, *retrieve, reuse, revise* and *retain* as illustrated in Figure 3.3.

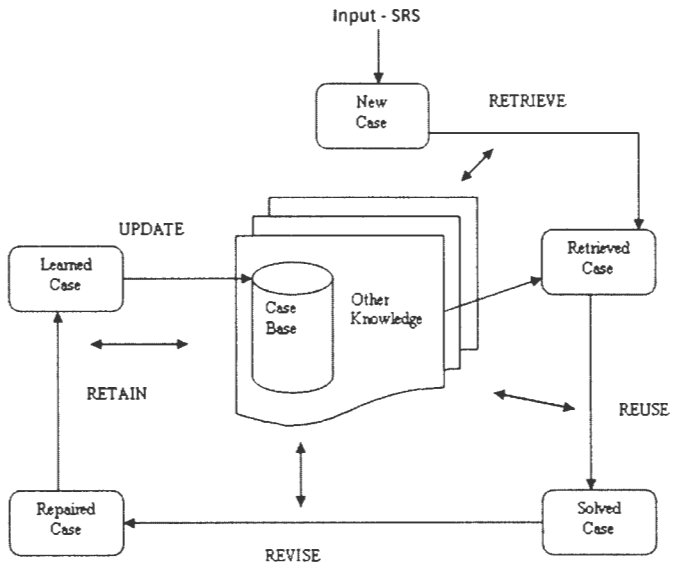


Figure 3.3: The CBR Cycle [*57*].

In Figure 3.3, a case is considered as an example of a previously experienced SRS in a CB. When a new case or problem is evaluated, previously stored cases are retrieved from the CB and are used to come up with the analysis results for the new case. If the new case (entered case) was experienced before, then the previously stored similar (exactly the same) case is retrieved and reused directly. Otherwise, the most similar case or a group of most similar cases are retrieved, revised and applied. Finally, the new case together with its solution is stored in the CB for future use [***49***]. Continuous use of CBR approach results in the CB growing bigger and bigger each time a new case is added, hence the need for a highly optimized tool to search through the cases and retrieve the most relevant case from the CB.

Considering Figure 3.3, CBR requires the use of effective information retrieval (IR) tools that matches and retrieve a case that is similar or closely related to the required query from a given repository. Bashir et al., discussed a number of IR techniques that can be applied by different systems [***58***]. These included content based image and video retrieval, grid enabled search in spatial and geospatial data as well as the e-learning multimedia retrieval technique. However, in our case we considered the Artificial Neural Network (ANN) due its learning and adaptive nature. This we found it suiting well with dynamic systems. ANN was introduced to measure the similarity level between the new and existing cases in the CB at the retrieval stage of the CBR cycle [***59***].

Figure 3.4, illustrates how the ANN is applied in case-based approach to match perceived new cases with the ones already in the CB. The ANN is represented inform of a compound equation where $Q_1 \ldots Q_n$ are the input values. $w_1 \ldots w_n$ are the assigned weights while v_i is the calculated input value and the associated weight which are further evaluated using an activation function [*57*], to give an output value which will be considered as the best matched case in the CB. When considering weights, it is important to ensure that the selected weights values always improve the performance of the ANN so that it remains optimal. In this direction, heuristics could be considered [*60*].

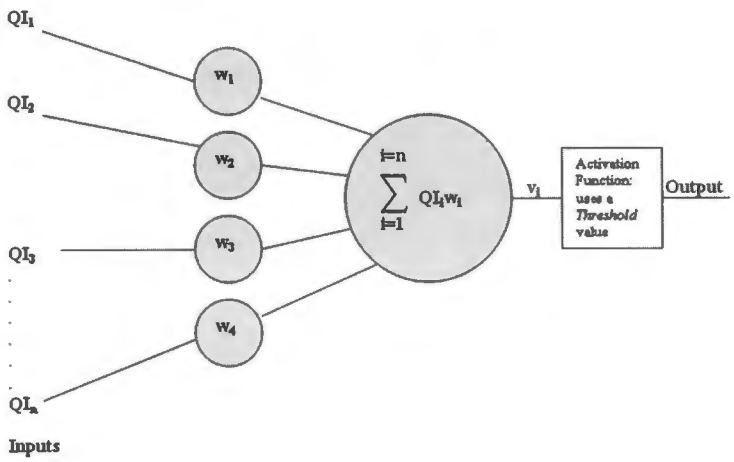


Figure 3.4: A Neuron of the ANN within the CBR Retrieve Stage [*57*]

3.6 Ontology-Based Requirements Analysis

In the area of knowledge engineering, researchers expressed different views on the definition of ontology, for example in [*61*], Lee and Zhao defined it as a *formal, explicit specialization of a shared conceptualization*, where conceptualization, explicit and formal refers to an abstract model elements that are clearly defined and that the model should be machine processable respectively.

However, as ontology is used to describe different domain areas Lee and Zhao reiterated the difficulty in providing a universal definition, hence described it as a set consisting of some core elements as presented in Equation (39);

$$O = \{C, R, H, T\} \quad (39)$$

In Equation (39), O is the ontology that can be expressed as a set of $\{C, R, H, T\}$ elements where C *and* R represents Concepts and Relations respectively, which are two sets that do not have anything in common (ie. do not intersect). H stands for hyperspace consisting of concepts and relations while T is the set of ontology theorems [***61***].

The Ontology approach can be split into three phases commonly referred to as definition of domain terms, decomposition of domain requirements and refinement and specification of domain requirements into primitive requirements using ontology.

3.6.1 Definition of Domain Terms

In this phase construction of domain terms that constitute a terminology dictionary take place. The dictionary evolves over time together with domain decomposition and this is meant to guarantee that the terms used are within the scope of the problem being defined throughout the project.

3.6.2 Decomposition of Domain Requirements

Once the terms in 3.6.1 are defined, the complex problem is decomposed using a subject-decomposition technique. The technique is based on modeling different subjective perspectives of different stakeholders such as users and developers on a system or a domain [***61***]. The subject-decomposition technique specifies its domain using formula 40;

$$D = (S, R, C) \quad (40)$$

Where S represent the viewpoints of the abstract stakeholders dealing with sub-problems, R representing relations of the elements of S and C being the constraints of S and R. Figure 3.5 shows an example of decomposition done on a generic Library Management System.

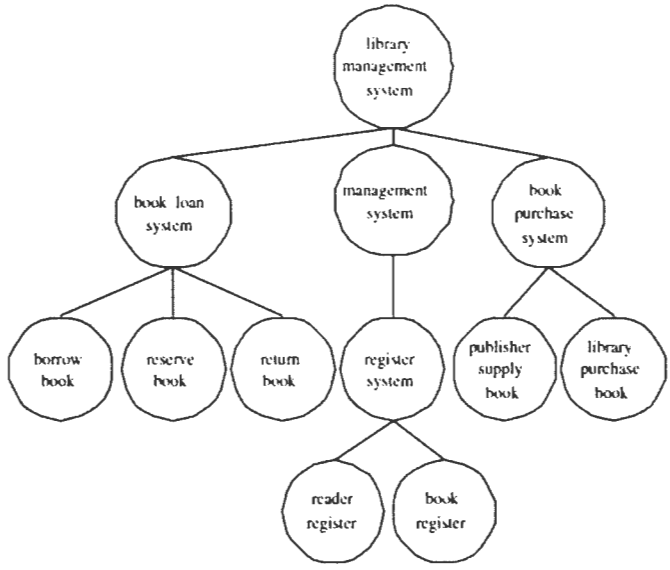


Figure 3.5: Library Management System Decomposition [*62*]

3.6.3 Refining & Specification of Domain Requirements using Ontology

After the decomposition phase, the requirements of each sub-problem domain are refined into functional and non-functional requirements and later primitive requirements [***61***] (i.e. atomic requirements that can't be further decomposed anymore).

Although some methods can be used to decompose complex systems into sub-systems, domain ontology is equally important to both stakeholders and engineers as an infrastructure to get requirements descriptions that are comparable. This is possible through the use of top-down functional, use-case and problem frame decomposition methods [***62***].

3.7 Factors to Consider when Selecting a RE Technique

Understanding the characteristics of RE techniques is crucial as it helps to identify the appropriate ones to be selected for a particular project [***63***]. To try and minimize the challenges faced in the selection process, Zhang [***64***], introduced some guidelines based on the four categories of elicitation methods which are *observational, conversational, analytical* and *synthetic*.

The study identified requirements *abstraction levels, sources, communication obstacles* and *levels of certainty* as perspectives that influence the selection process. For example in [*46*], [*47*] the findings were that observational techniques support RE process well if the objective is to analyze the problem (requirements' abstraction level) of a new domain (level of certainty) from sources other than human beings (requirements source) especially in an organization that has a strong culture (communication obstacles).

While conversational approaches such as interviews are quite suitable in situations where we need to acquire comprehensive information and knowledge from stakeholders, whereas scenario-based analysis respond well in situations where we need to comprehend contemporary work of the systems under study. User stories and workshops perform better in gathering visions from end users and in-depth discussions respectively [*65*].

Although different researchers justify their approaches as most suitable, it however remains apparent that analysts should be aware of the factors they should consider and their effects before concluding. Figure 3.6 illustrates four major factors that primarily affect the RE techniques which require the attention of the analysts in their process of selecting the best technique to implement.

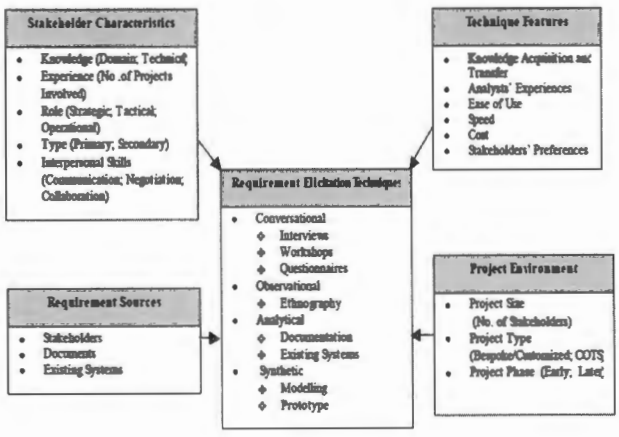


Figure 3.6: Factors Affecting the Selection of RE Techniques [*64*].

In Figure 3.6, the illustrated analytical techniques encompass requirements reuse, documentation, repertory grid, laddering and card sorting as discussed earlier, while synthetic techniques include but not limited to prototypes, contextual inquiries, scenarios/ storyboards.

This technique sometimes combine conversation, observation and analytical approaches systematically into a single method known as collaborative where analysts and stakeholders cooperate in diverse ways to reach common a understanding.

In RE, selecting the best technique would mean choosing one that facilitates knowledge acquisition and transfer between users and the analysts in a smooth manner. Anwar and Razali did a survey on how each of the four major factors illustrated in Figure 3.6 affect the RE selection process [*63*].

For example, under Technique Features, Table 3.1 shows results from a survey conducted on how some software development experts rate the effects of knowledge acquisition and transfer, analysts' experiences with a particular technique, ease of use, speed, cost and stakeholders' preferences in selecting RE for a particular project. The first two factors were found to be the most influential with the highest frequency.

Technique Category	Factor	Frequency (No. of Informant)
Conversational: Observational: Analytical: Synthetic	Knowledge Acquisition and Transfer	5/5 (100%)
	Analysts' Experiences	5/5 (100%)
	Ease of Use	3/5 (60%)
	Speed	2/5 (40%)
	Cost	2/5 (40%)
	Stakeholders' Preferences	1/5 (20%)

Table 3.1: Selection Factors Based on Technique Feature [*63*].

3.8 Performing RE for Dynamic Systems

Most RE models focus on specific methodologies or techniques. For example the Robertson's Volere requirements methodology includes a detailed process model of its RE activities with inputs, outputs and recommended techniques for each activity [*66*]. Given the number of models of RE that are in place, we found out that each model was equally important at a given stage of the elicitation process hence the need to have these models harnessed. Here we discuss in brief one model of elicitation that represents a generalization of all known elicitation methodologies suitable for dynamic systems known as the Generalized Model.

3.8.1 The Generalized Model

Hickey and Davis described elicitation process as a series of activities that can be represented using a mathematical function as in Equation (41) [*66*].

$$elicit_i = (R_i, S_i, t_i) \rightarrow R_{i+1}, S_{i+1} \tag{41}$$

where i is the elicitation process step, $t_i \epsilon T$ a set of all known elicitation techniques and S_i and R_i are the current situation and state of knowledge of the requirements that needs to be understood respectively. R_{i+1} and S_{i+1} are the new state of the requirements and situation derived from the previous conditions. From equation 41 it follows that for every RE methodology say M_j with n steps can be represented using a mathematical function as follows;

$$R_n(M_j) = elicit_n(\dots(elicit_2(elicit_1(R_1, S_1, t_{j1}), t_{j2})\dots), t_{jn}) \tag{42}$$

where R_1 and S_1 are the initial requirements and situation state respectively while $t_{j1}, \dots t_{jn}$ are the steps prescribed by the methodology M_j [*66*]. Equation 42 brings together all the relevant RE techniques in an ordered approach whereby output from one technique serves as input into the next technique. This approach is very important in that it exhaust all the strength found in different techniques in one process.

3.8.2 Case Study – A RE Approach for the GenApp.

The purpose of this case study is to describe a RE approach that we consider for the development of the GenApp. In Figure 3.7 we describe the basic processes in the cycle of the approach.

The starting point of RE cycle was an informal problem description done in step 1 and its goal was to collect and structure knowledge about current algorithms and/or methods used for dynamic parameterization and their roles in the systems they interact with, which was achieved by reading and reviewing related documents published by different authors.

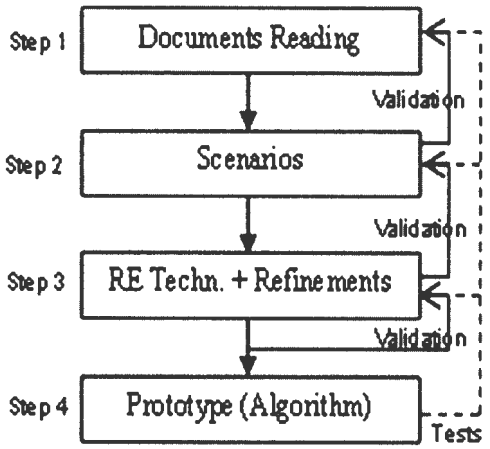


Figure 3.7: Requirements Elicitation Cycle

In the articles we reviewed, such as Moreno et al., discussed a strategy that selects the filter parameters such as the (orientation ($\emptyset$), aspect ratio (γ) and sigma (σ)) semi-automatically using the Information Diagrams (ID) strategy while the rest of the parameters (wavelength (λ), phase offset (φ), bandwidth (β)) remained fixed and were specified based on empirical data [*10*].

Step 2 looked at developing scenarios that exposes the current systems' strengths and weaknesses aimed at further specification of the problem under review. At this stage the Gabor Filter function (GF) was simulated using MatLab and evaluated using different scenarios. This was meant to ascertain the challenges faced by FET with parameterization.

In Figure 3.8 (a) we illustrate an input image that was used during our simulation process and Figures 3.8 (b) – (d) represents the results obtained from the simulated GF filter with different parameter scenarios. The results obtained from the simulations done shows that Figure 3.8 (d) is somehow close to the input image Figure 3.8 (a) after filtering as compared to Figures 3.8 (b) and (c). However, the results in Figure 3(d) were obtained after several attempts using different parameter value combinations, a hit and miss scenario.

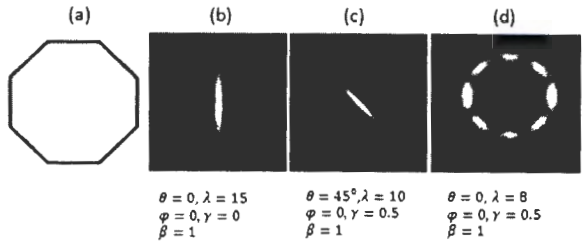


Figure 3.8: Results Obtained from the Scenarios on MatLab Simulation of the GF Function.

Step 3 involves abstraction of the scenarios presented in stage 2. RE techniques discussed in sections 3.2 to 3.5 were selected and applied of which a combination of the use-case based and case-based requirements analysis techniques (3.3 and 3.4) produced a better scenario that defined the interaction between external actors and subject system in addition to reuse of previously acquired knowledge. Combining the efforts of the two techniques resulted in an approach regarded as the collaborative approach.

Step 4 looked at building a prototype that summarizes the problems described and specified in the first two stages and a solution part with functional, non-functional requirements, cases and scenarios. However, we found out that we still needed to support iteration of the requirements from one stage to the other; hence we introduced Equation (42), as discussed in section 3.8.1 in a way that we could obtain new requirements from the coalition of current and previous requirements at each iteration step.

3.9 Chapter Summary

The chapter looked at different approaches to RE and factors that should be considered if we are to select an appropriate approach for a given problem. A case study was done to describe and validate the RE approach for dynamic parameterization which was presented in four steps showing the processes involved at each step. The idea was to use this approach in the development on the GenApp.

It was quite important to observe that due care must be exercised when applying RE methods given that their suitability depends on the application areas and scenarios presented, hence the need to harness them where possible to address specific needs. In this case the Generalized

Model worked well in the requirements elicitation process for the development of the GenApp.

Chapter 4: The GenApp: Novel Adaptive Algorithm

4.0 Chapter Overview

In this chapter we proposed an approach named the *"GenApp"*, derived from *Genetic Approach*, which is our novel adaptive algorithm that implements genetic algorithms (GAs) to determine parameter values for FET using the adaptive principles of the biological genes. In feature extraction there are many areas of specialization that can be looked into, for example some algorithms have been designed to identify specific target objects while others focus more generally on say buildings or roads extraction and the extraction techniques for these different specializations can vary substantially.

The background study focused on the algorithm development process and the three basic genetic operators that we used to implement the GenApp, which are the selection, crossover and mutation. The GenApp flowchart and algorithm were presented as our deliverables from the chapter. The authors acknowledge that most of the work in this chapter was published in a peer reviewed journal, *The International Journal of Soft Computing and Engineering (IJSCE)* ISSN: 2231-2307, Volume-4, Issue-2, May 2014.

4.1 Background Study

Genetic Algorithms are an adaptive heuristic search algorithm found on the evolutionary ideas of natural selection. They follow some sequence of processing through a finite number of iterative operations implemented using genetic operators.

4.1.1 The Genetic Operators

Genetic operators were first proposed by Holland as directed random search techniques which can find the global optimal solution in complex multi-dimensional search spaces [*67*]. They are known to employ different genetic operators to manipulate individual solutions in a given set of solutions (population) over several iterations (generations) to gradually improve

their fitness. This has made them to be successfully applied in many engineering and optimization problems.

There are three basic genetic operators that are used to generate and explore the neighbourhood of a population and select a new generation. These are *selection, crossover and mutation* [***67***]. The operators are applied after an initial population of say, ***N*** solutions has been generated and each solution ***S*** in the currently generated population is evaluated using a fitness value obtained using equation (43).

$$F(s) = 1.5 * SqE_{max} - SqE(s) \tag{43}$$

Where, $F(s)$ is the fitness value of solution S, SqE_{max} is the maximum square error in the current generation and $SqE(s)$ is the current square error (SqE). The SqE is obtained by computing the following equation;

$$SqE = [(\sum_{n=1}^{N}\sum_{d=1}^{D} X_{nd}^2) - (\sum_{k=1}^{K}\frac{1}{Z_k}\sum_{d=1}^{D} SF_{kd}^2)] \tag{44}$$

Where, SF_{kd}^2 is the square of the sum of the d^{th} solution, K is the number of clusters that are used to group chromosomes, N is the number of patterns represented in vector form and D represent the vector dimension.

At this stage different selection methods such as the *stochastic* or *ranking based selection* are used to select individual solutions with a high fitness value from the selected population.

The *crossover operator* work with pairs of selected individual solutions from the population and is defined in different ways. Given, for example a pair of binary coded solutions **111111** and **000000**, the crossover operator could mean swapping only a single bit between the two solutions commonly known as *one-point crossover* (Figure 4.2) or swapping several bits in the binary solutions at once (Figure 4.3).

Figure 4.1 illustrates a general overview of the individual solutions before, during and after the implementation of the crossover operator.

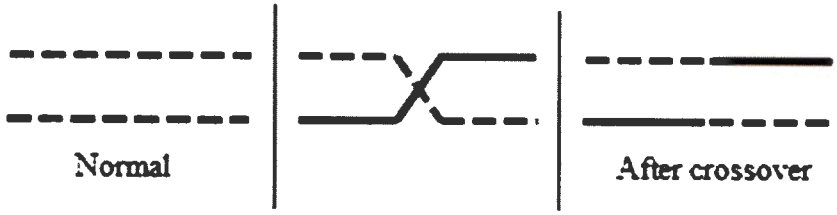


Figure 4.1: The Crossover Operator Overview

Figure 4.2: The Single-bit Crossover

Figure 4.3: The Multiple-bits Crossover

From Figures 4.1, 4.2 and 4.3 we can relate that though it's subjective to the selected initial population. The crossover operator can generate large amounts of different possible solutions which may not be quite close to the expected optimum solution or might not cover enough the entire GA's problem space. As a way to minimize the gap, a *mutation operator* is performed either on the selection or crossover stage. This will see the elements of the binary solutions being altered from $\mathbf{1}^s$ to $\mathbf{0}^s$ or vice-versa. The mutation operator is implemented using a probability that is kept very low (i.e. 0.001%) as high mutation might end up destroying fit strings (binary solutions) and degenerate the GA into a random walk.

Figure 4.4 shows a flowchart of a simple GA. The algorithm iterates until a predefined number of generations has been reached.

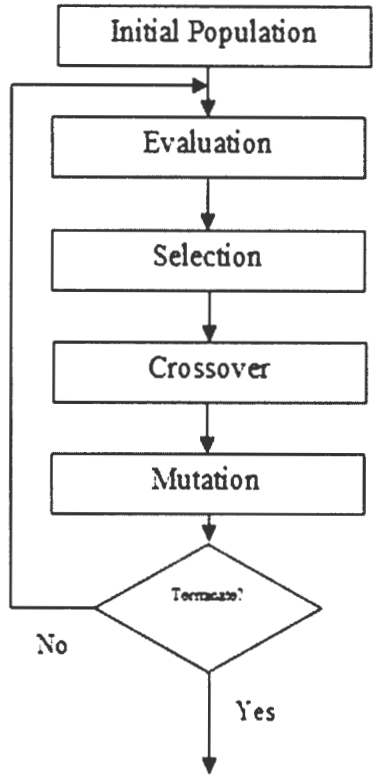


Figure 4.4: Flowchart of a Simplified Genetic Algorithm [*67*]

The distinguished feature of the GenApp is that it initially generates values for the initial population from a range specified by Information Datagram (ID) strategy and then uses these values with the same weight to find a good solution, while GAs operators such as mutation and crossover follows to generate permutations of different weights in order to improve the first computed relatively *"good"* solution. In this case the GF that we discussed in section 2.5.2 was considered as our control to benchmark our algorithm.

The characteristics of the GenApp are input, output, definiteness, finiteness and effectiveness. In developing our algorithm, we considered different algorithmic techniques commonly used such as the Divide and Conquer and Dynamic Programming.

The phases involved in the algorithm development process are shown in Figure 4.5, which are the design, analysis and implementation. The design and analysis are iteratively repeated until we achieve a time and space efficient algorithm.

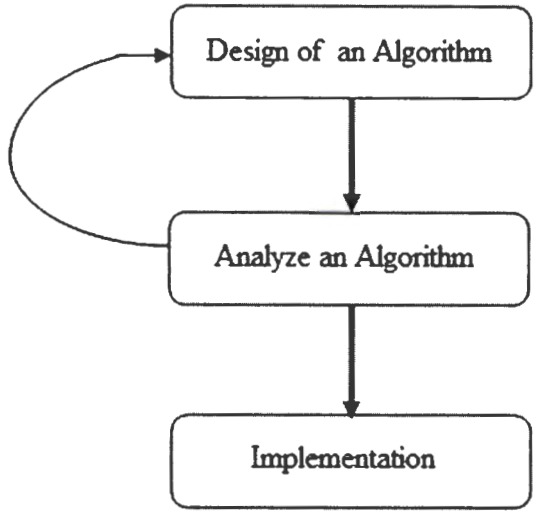


Figure 4.5: Algorithm Development Process Phases [*68*]

Once the first two phases are completed, the process moves towards the implementation of the algorithm as the last phase.

4.2 The *GenApp* Analysis and Design

The principal idea of the GenApp is to initially select and maintain a population that cover a bigger part of the GA's problem space. The algorithm iterates in such a way that at each process step the algorithm evaluates the current set of solutions known as current generation and chooses the best of them to serve as the *"parents"* for the new generation. The iterations are realised until either a *"good"* solution is found or the number of iterations exceeds a specified value otherwise no further improvements to the current solution are recorded.
The following key issues were considered in our analysis and design of the algorithm, that is;

(i) The size of the initial population, given that a bigger population size ensures sufficient diversity in solution selection.

(ii) How the solutions are represented so that they can be computed using our formulae such as fitness value and mean square error, equations 27 and 28 respectively.

(iii) Which fitness function to apply when evaluation a given solution.

(iv) How to manipulate the representation of the current solution to obtain a newly improved solution, the crossover operation.

(v) The criterion to use for choosing individual solutions that will serve as "fit parents" for the next generation, the selection operation.

(vi) The approach to avoid convergence of a solution to a set of equal solutions, the mutation operation.

Considering the above key issues, we managed to come up with the following processes as represented in the flowchart as shown in Figure 4.6.

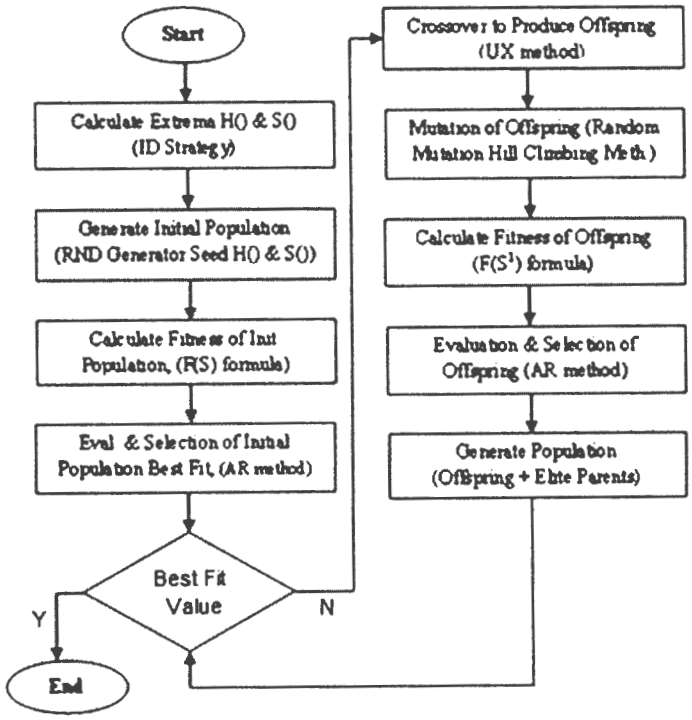


Figure 4.6: Proposed GenApp Flowchart [*69*].

4.3 The GenApp Implementation

The proposed Genetic Approach (GenApp) is principled on the GF technique discussed in section 2.5.2. It strives to optimise the results obtained by the GF through its adaptive

parameter selection scheme inspired by the adaptive principles of the biological genes that we modelled using Genetic Algorithms (GA). The GenApp extended the ID strategy and used it to determine the initial ranges of the parameter values. The ranges are denoted as extrema, that is highest and smallest extrema representing the upper and lower limits respectively. The approach helps determine parameter values for the major GF parameters such as $(\sigma, \gamma, \theta, \sigma_x, \sigma_y)$. Using GenApp we propose that parameter θ for example, be specified using as in algorithm 1;

Algorithm 1: Specifying Parameter θ

---

1. *Divide the input image into blocks of workable sizes* $(W \times W)$ *and calculate the local gradients* G_x *and* G_y *at each pixel* (x, y) *in each block.*

2. *Estimate the local orientation of each block using formula 7.*

3. *Calculate the* θ*-ID extrema, Highest Local Maximum* $HLM(\theta)$ *and Smallest Local Minimum* $SLM(\theta)$ *using equations 20-25.*

4. *Generate values* $I_\theta = \{\theta_1 \dots \theta_n\}$ *for the initial population such that* $SLM(\theta) \leq I_\theta \leq HLM(\theta)$.

5. *Calculate the average rank* AR_θ *for each* I_θ *value as follows,* $AR_\theta = \sum_I (AR_m^I)\ /n$ *where m is the population member, n is the total number of population members and I is the population dataset.*

6. *Apply uniform crossover (UX) method with a 0.5 probability and mixing ratio.*

7. *Apply Random Mutation Hill Climbing (RMHC) method with a probability of 0.01.*

8. *Repeat (5) through (7) until you obtain the required threshold fitness value or until you reach the number of specified generations.*

Implementing this algorithm would help approximate a value that is likely to give a better output image from the FET. The rest of the parameters can have their values computed using the same algorithm (algorithm 1) with a few relative variations especially on steps (1), (2)

and sometimes (3). For example the deviations σ_x and σ_y instead of empirically setting them to a value, say σ_y being set to 4.0 or 3.0 as indicated in other literature. Using the GenApp the values of the parameter can be specified using algorithm 2.

Parameter σ_x's value is directly related to the chosen period T in GF or T_1 and T_2 in MGF. As discussed in [*17*], it is quite important and mostly recommended that we first compute the periods given that the larger they are, the wider the expected bandwidth whereas a too wide bandwidth can unexpectedly enlarge the noises. Again, a too narrow bandwidth tends to suppress some useful signals such as colour intensity of the original image, which is required for image processing.

Algorithm 2: Specifying Parameter σ_y

---

1. *Set the value 4.0 as the initial estimated value of* I_{σ_y} *then specify a range such that* $SLM(\sigma_y) \leq I_{\sigma_y} \leq HLM(\sigma_y)$.

2. *Apply steps (4) through (8) of algorithm 1.*

The first three stages of the approach (Initial population, Evaluation & Selection) use *phenotype representation* which is the physical appearance of the individual possible solution. While the crossover and mutation stages are implemented using *genotype representation* which is the binary representation of the phenotype. The genetic operators at each stage of figure 4.4 are implemented as follows;

4.3.1 Implementation of Genetic Operators

The purpose of this phase was to customize the generic genetic operators so that they address the requirements of the GenApp. This was done by looking at each genetic operator and assigning an appropriate algorithm that best fit the implementation of the GenApp as discussed in sections 4.3.1.1- 4.3.1.4.

4.3.1.1 The Initial Population

The initial population was generated from the values specified by the ID strategy for the case of $\emptyset$, $\boldsymbol{\sigma}$ and $\boldsymbol{\gamma}$, where we consider the local extrema of each parameter as illustrated below;

(i) For each selected parameter say $\emptyset$, we first compute local extrema, that is, the highest local maximum $H(\emptyset)$ and smallest local minimum $S(\emptyset)$ using the ID strategy.

(ii)Specify the local extrema $H(\emptyset)$ and $S(\emptyset)$ as the upper and lower limits of the initial population for the approach.

(iii) Randomly generate values from within the extrema range $[S(\emptyset), H(\emptyset)]$ and consider the generated values as the initial population for the GenApp.

(iv) Using the fitness formula (43), calculate the fitness value of each member as part of the evaluation process of the initial population.

(v) Using a selection method described in 4.3.1.2, select the highly fit members from the current population that will constitute the next population.

Once the above is achieved, we are guaranteed of an initial population with most of its members having favourable SqE and $F(s)$ values.

4.3.1.2 Selection

At the selection stage, the GenApp used the Average Ranks (AR) ranking method to select possible solutions in the current population that will be passed to the next population. Since the approach used the minimum SqE technique, we therefore expected members with a small SqE value to have a higher possibility of surviving selection in the current generation [*70*].

Based on equation 45 the AR generated values that were assigned to the current population members and this helped us to properly select the best fit solution, we ordered the members

according to the measured average rank $\boldsymbol{AR_j}$ assigned to each member and the following steps illustrate the operation of the AR ranking method;

(i) Let AR_j be the average rank for each member in a generation, calculate the AR_j as:

$$AR_j = \left(\sum_i AR_j^i\right)/n \tag{45}$$

Where i is the dataset, j is the population member, AR is the rank and n is the population in a generation.

(ii) Arrange the average ranks (AR_j) for the members in the current population in ascending order.

(iii) Assign ranks starting with the least AR_j value, such that the best fit member is the one with the least AR_j value and assign its rank as 1, the second gets rank 2, and so forth.

After this stage there was a high probability that the members with the best rankings are closer to the parameter value we were searching and the major function of the remaining genetic operators (crossover and mutation) was to optimize the final solution.

4.3.1.3 Crossover Operator

The crossover operator mimicked propagation and here we implemented it by crossing pairs of chromosomes to generate new offspring using the uniform crossover (UX) with a mixing ratio of 0.5. In our approach, the mixing ratio was arrived at based on empirical experiments and analogy that in a large search space, a GA that uses UX outperforms a GA that use one-point or two-point crossover [*71*].

Figure 4.7 confirms the efficiency of the UX over the one-point and two-point crossover as stated earlier, though the experiments carried out indicate that the UX outperforms all other crossover operators mostly in a larger population size.

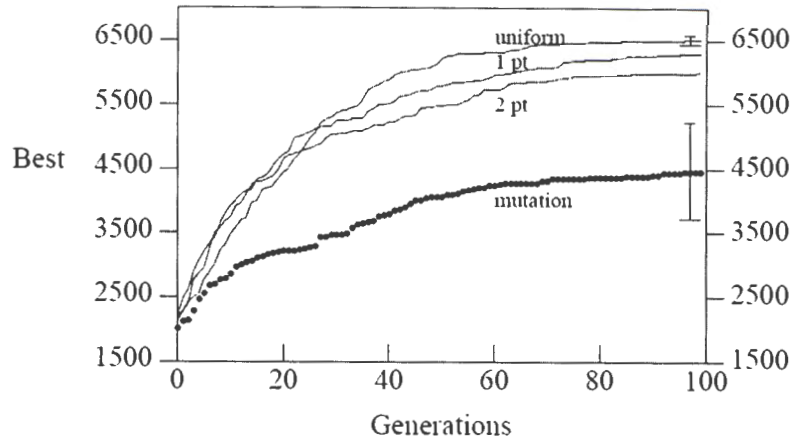


Figure 4.7: Performance Test of Crossover Operators With a population size 100 Using GenNet [*71*].

By using a mixing ratio of 0.5, it follows that the generated offspring has approximately 50 percent of its genes from the first parent and the other 50 percent from the second parent, even though the crossover points are randomly chosen as shown in Figures 4.8 (a) and (b).

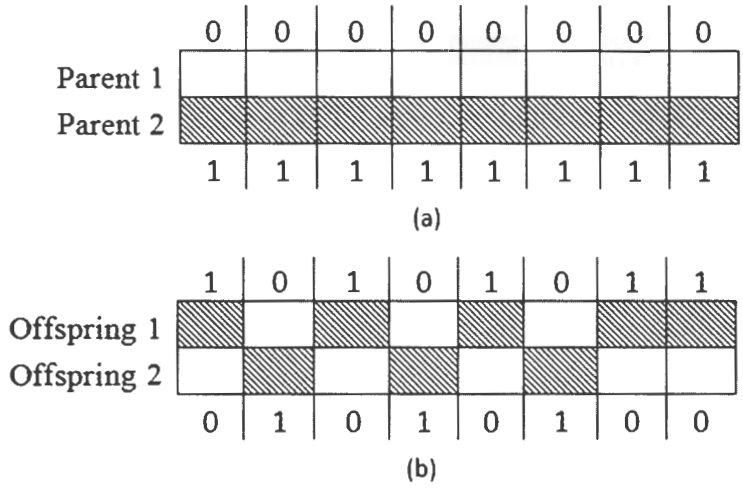


Figure 4.8: Uniform Crossover with a Probability Ratio Of 0.5, (a) parent chromosomes, (b) offspring after mating with 50% genes from each parent [*69*].

In our approach we expected the crossover operator to contribute 50 percent of the population for the next generation, while mutation and the "*elite*" chromosomes (parents with best fitness value in current generation) to contribute 20 and 30 percent respectively.

4.3.1.4 Mutation

Unlike the crossover, the mutation operator mimics the random changes in nature of the chromosomes within a generation. Since our initial population was derived from a controlled search space, we only consider it for local optimization that is, refining the already generated solution.

As a result, we considered the Random Mutation Hill Climbing (RMHC) to randomly select a neighbor for a candidate solution and mutate only if it improves the current results, otherwise the operator remains unexecuted. Using the RMHC the complete set of features was represented by a binary string of length *N*, where a bit in the string is set to "**1**" if it is to be kept and set to "**0**" if it is to be discarded [*72*]. The following steps summarises the RMHC method.

(i) Initialize a binary string S, of length N, where M features are marked as used, '1' and the remaining N-M are '0'

(ii)Convert the binary string S to phenotype S^l and test its fitness $F(S^l)$ using a fitness function (8).

(iii) Randomly mutate M bits in the binary string S.

(iv) Return to step (ii) and continue until either the fitness goal is reached or the maximum number of iterations is reached.

4.3.2 The GenApp Algorithm

Combining the stages discussed in section 4.3.1.1 through to section 4.3.1.4, resulted in our algorithm as listed below.

Algorithm 3: The GenApp

---

1. *Initialize ();*
2. *Compute Extrema (H, S) using ID Strategy (equations 22 -27);*

3. *Generate* (P_c), *where* ($S \leq P_c \leq H$);
4. *Compute Fitness (F(s) equation 43)*
5. *Evaluate* (P_c); *// AR ranking method*
6. *Res=BestOf* (P_c);
7. *Repeat_Loop1 until Stop (Criterion)*
8. $P_n = 0$;
9. *Repeat_Loop2* (P_c)*/2 times // UX ratio of 0.5*
10. *Select* $P_1 \in P_c, P_2 \in P_c$;
11. *Crossover* (P_1, P_2, C_1, C_2);
12. $P_n = P_n \cup \{C_1, C_2\}$;
13. *End Repeat_Loop2;*
14. $\forall p \in P_n$, *Compute Fitness (F(s));*
15. P_c = *BestOf* (P_c) $\cup$ *BestOf* (P_n);
16. *Evaluate* (P_c);
17. $\forall p \in P_c$, *Possible Mutate (p) //RMHC method with a probability of 0.01*
18. *Evaluate*(P_c)
19. *Res = BestOf*($P_c \cup \{Res\}$);
20. *End Repeat_Loop1*
21. *Output (Res);*

In the above algorithm (steps 1-21) the genetic representation of each individual is formulated by *Initialize ()* and the current population P_c is constructed from randomly generated individuals by *Generate* (P_c). Compute *Fitness (F(s))* uses the SqE technique to calculate the fitness of each individual member P, Evaluate (P_c) assign fitness ranks to the individuals, and *BestOf (P)* finds the individual with the highest fitness value.

Repeat_Loop1 simulates generation cycles while *Criterion* terminates the simulation either through set number of generations or when optimized solution is reached and *Repeat_Loop2* simulates the UX operation. In each generation, a set of offspring P_n of size P, was yielded by the crossover operation *Crossover* (P_1, P_2, C_1, C_2) where P_1 and P_2 are the mating parents yielding two offspring C_1 and C_2 which are added into the current population P_c. *Mutate (P)*

uses a *0.01* probability to randomly change the genotype of each individual in P_c while *Output (Res)* gave the final result from the algorithm.

4.4 Chapter Summary

The chapter looked at developing the novel adaptive algorithm named the GenApp. The development process followed the contemporary algorithm development phases as outlined in Figure 4.5. The fascinating features of the algorithm were the inclusion of the information datagram concept and the fusion of the genetic operators into algorithm. The two features were meant to screen the initial population to serve the algorithm from going through several iteration steps, select and optimize the parameter values respectively.

We deliberately kept the mutation percentage at 0.01% to avoid huge diversions from the solution thereby reducing processing time per generation and uniform crossover was preferred to avoid bias towards one solution set during mating. Later in the chapter we delivered the GenApp flowchart and algorithm.

Chapter 5: Simulation and Performance Evaluation of the GenApp

5.0 Chapter Overview

This chapter aimed at evaluating the performance of the GenApp algorithm, in terms of efficiency and make recommendations on its use. This was done by looking at similar algorithms as discussed by different authors in view of qualifying our own. In this regard, the GF, MGF approaches introduced by Yiang et al in [*17*], the PCA, LDA, ICA that were discussed by Jeyabharathi et al. in [*73*] were considered as our performance control.

The "*Big Oh notation*", an algorithm evaluation technique and fitness value tests were performed on the GenApp and the results were analyzed as part of the evaluation process. The results obtained from our experiments reflected a low complexity on the execution algorithm, reduced number of generations in finding an optimum parameter value and a relatively constant fitness value, which to some extent gives us confidence in the algorithm's potential to improve results obtained from FET.

Simulations on the performance of the GenApp were done using genAID, a simulation tool that uses macros in standard Microsoft Excel, coded in Visual Basic language. We incorporated Microsoft excel worksheets to capture data for our simulations and the data we worked with was presented in both genotype and phenotype formats.

5.1 Background Study

The background study reviewed some of the related work on analysis and performance evaluation of the aforementioned FET. The review was based on experiments performed by different authors as discussed in their publications.

5.1.1 The TGF and MGF

The TGF and MGF as discussed in [*19*] were analyzed using some fingerprint image extracted from an image database captured using an optical live-scanned equipment. The tests made a comparative analysis on the robustness and efficiency of the two algorithms.

The experimental results reviewed, showed a great improvement on the quality of the output image for the MGF and the Open-Close Sequence (OCS). This could be attributed to an improved parameter selection strategy for both the MGF and the OCS.

5.1.1.1 Testing For Robustness

Figure 5.1(a) shows an original fingerprint image taken from the database, while 5.1(b) and 5.1(c) are the results obtained using the TGF with the values $\sigma_x = 2.0$ and $\sigma_x = 2.5$ respectively. These Figures represent the best parameter values obtained using the TGF approach. Parameter σ_y was empirically assigned a fixed value of 4.0 throughout the experiments on both approaches.

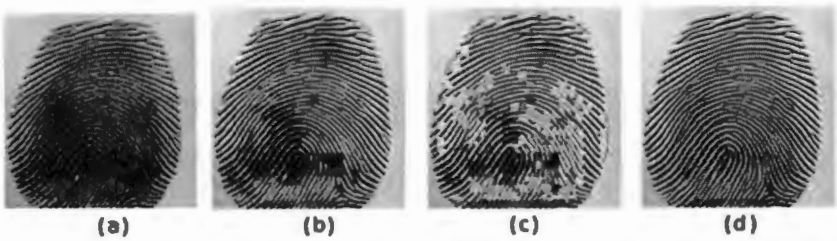


Figure 5.1: Enhancement Results Corresponding To The Fingerprint Image (a) [*19*].

Figure 5.1(d) shows the resulting fingerprint image obtained using the MGF. Looking at the fingerprint images, it is evident that the Mahalanobis distance between 41(a) and 5.1(d) is almost insignificant as compared to 5.1(a) and 5.1(b) or 5.1(c)

5.1.1.2 Testing For Efficiency

A Pentium 4 machine with a 1.3 GHz processor speed and 128 Mb of RAM was used to test the processing time (in seconds) for both algorithms. Different image resolutions were

selected for each test run with a fixed σ_y value of 4.0 as previously stated and a convolution of 11 for the TGF as shown in Table 5.1.

Although the results in Table 5.1 show a slightly higher average computational cost for the MGF in each test run (circled), this can be attributed to a larger MGF convolution mask of $\left(\frac{T_2}{2}+\frac{T_1}{2}+\frac{T_2}{2}\right)$ as compared to the TGF of $(2N+1)$. Using a larger mask size gives an added advantage in that errors emanating from the effects of noise are reduced by local averaging within the neighbourhood of the mask.

Image resolution (pixel)	TGF ($N = 5$, $\sigma_y = 4.0$) σ_x	Average time cost (s)	
		TGF	MGF
224 × 288	1.8	0.90	0.92
256 × 256	1.8	0.94	1.01
364 × 256	2.0	1.13	1.22
400 × 376	2.2	1.76	1.89

Table 5.1: Comparison of time cost (in seconds) between the TGF and MGF [*19*]

Secondly, the performance of MM based algorithms as discussed in [*74*] is analyzed using a well-known corrupted image called Lena. The tests were carried out by making a comparative analysis of the probabilities of the pixels that were correctly detected against those wrongly detected by the algorithm.

5.1.2 The Mathematical Morphology

A novel filter Open-Close Sequence (OCS), presented by Ze-Feng et al. in [*74*] that is based on mathematical morphology for high probability impulse noise removal was tested using simulation experiments to demonstrate its performance. Comparisons were made with other similar filter techniques using a well-known *"Lena"* image that was corrupted by salt and pepper noise with equal possibilities.

5.1.2.1 Testing For Efficiency

The efficiency (γ) of the detectors that implement the algorithms is calculated using the following formula;

$$\gamma = (1 - \alpha) \times (1 - \beta) \tag{46}$$

Where α is the probability of corrupted pixels that were missed and β being the probability of pixels that were falsely detected. From equation 46, it's worth noting that efficiency in this case is the product of the probabilities of corrupted pixels that were detected with the pixels that were correctly detected using the algorithms.

size of b		3X3	3X3	3X3	5X5	5X5	5X5	7X7	7X7
T		0	100	200	0	100	200	0	100
M	α	0.0000	0.1020	0.3820	0.0000	0.0031	0.0152	0.0000	0.0000
R	β	0.2743	0.1476	0.0163	0.0044	0.0037	0.0004	0.0001	0.0001
D	γ	0.7257	0.7655	0.6079	0.9956	0.9932	0.9844	0.9999	0.9999

Table 5.2: Effects of the size of the SE on output image [*74*]

Table 6 shows the effects on the size of a chosen structuring element b to the overall quality of the image being extracted. For example, the table shows efficiency values of 0.9999 and 0.6079 corresponding to the 7x7 and 3x3 SE respectively. Meaning to say best values were obtained from the 7x7 SE.

In this case we can ascertain that the larger the SE the better output image produced as the probability value of 0.9999 is evident enough that the extracted image is very close to the original image which has a probability value of 1.

5.1.2.2 Testing For Robustness

Further experiments were reviewed which tested the robustness of the MM algorithm, the OCS algorithm with other algorithms that are used in nonlinear filtering techniques such as

the Pixel-Wise Morphological Anomaly Detector (PWMAD), Network Adaptive Switching Median (NASM) and the Standard 5x5 Median.

From Figure 5.2 it can be seen that the OCS performed robustly across all the noise ranges from 10% to 80%. The rest of the algorithms fell abruptly as the noise ratio went above 40%. This is evident that the rest of the algorithms reviewed are able to produce meaningful results only on slightly corrupted images.

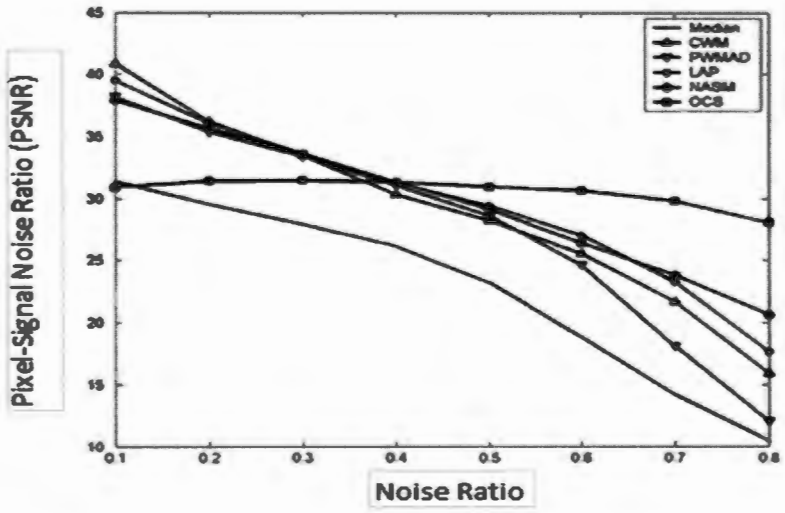


Figure 5.2: PSNR values for different filter algorithms operating on the image "Lena" [*74*].

5.1.3 The Principal Component Analysis

Jeyabharathi et al. in [***28***], analyzed the performance of PCA, Linear discriminant analysis (LDA) and Independent Component Analysis(ICA) using the Recognition Rate (RR) and F-Score (FS) as performance metrics. In their experiments they showed the performance results of the three FET in recognizing features that were provided and their ability to combine with other algorithms such as the Support Vector Machines (SVM) and Nearest Neighbour (NN). Table 5.3 shows the test scores obtained from the experiments while Figures 5.3 and 5.4 show the overall performance of the three FET using the RR and FS respectively.

Classification Methods	SVM Classifier			NN Classifier		
Metric	Recognition Rate					
Images\Algorithms	PCA	LDA	ICA	PCA	LDA	ICA
Train	90%	60%	90%	38%	44%	52%
Ship	100	80%	60%	34%	36%	40%
Bike	100	90%	50%	32%	36%	76%
F Score						
Train	0.6	0.4	0.6	0.25	0.29	0.35
Ship	0.67	0.46	0.4	0.23	0.24	0.27
Bike	0.67	0.46	0.33	0.21	0.24	0.50

Table 5.3: Performance Analysis Based on RR and FS Metrics [*28*]

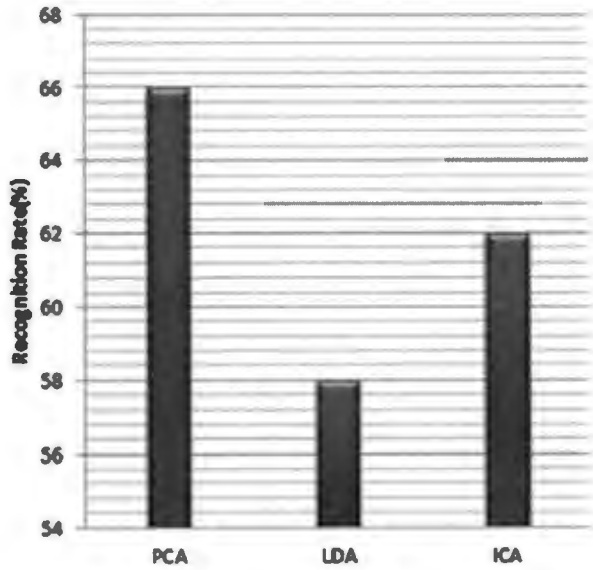


Figure 5.3: Performance Analysis Based on RR [*28*]

The RR presented in Figure 5.3 measures the ability of the FET to recognize the correct and false features of a given image. It is measured using Equation (47);

$$Recognition\ Rate = \frac{Number\ of\ recognized\ images}{Number\ of\ testing\ images} \tag{47}$$

While FS measure illustrated in Figure 5.3, is a measure of the test's accuracy by considering the precision p and recall r of the test. It is measured using Equation (48);

$$F - Score = 2 \times \left(\frac{p \times r}{p + r}\right) \quad (48)$$

Where p is the number of correct positive results divided by the number of all positive results obtained in a test and r is the number of correct positive results divided by the total number of expected positive results.

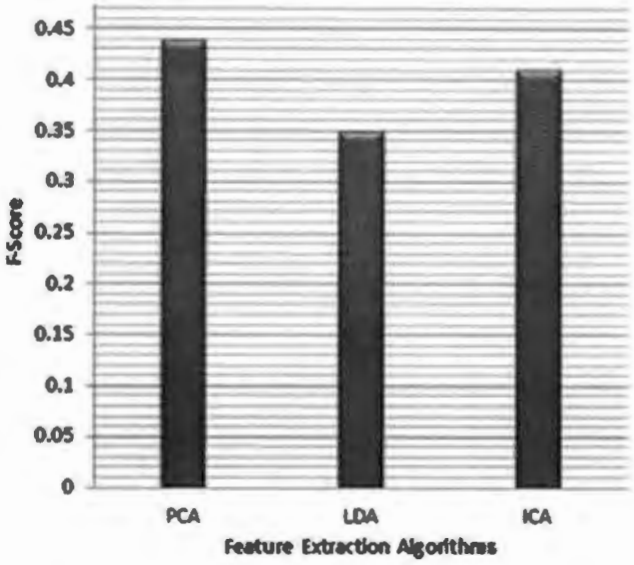


Figure 5.4: Performance Analysis Based on FS [*28*]

From the results presented in Figures 5.3 and 5.4, it can be noticed that the PCA managed to provide more accurate results as compared to other two techniques.

5.1.4 The Scale Invariable Feature Transform

Khan et al. in [*75*] discussed the performance of SIFT and Speeded Up Robust Features (SURF) against various deformations benchmark dataset. A naive algorithm was used to measure their performance on an image matching mask that was selected from a set of datasets. They performed various experiments to evaluate the matching performance of all proposed descriptors on image transformations (rotation, illumination, and noise invariance), blurring (scale and viewpoint) and general matching on execution time. Figure 5.5 and Table 5.4 illustrates the performance of different SIFT method variations with SURF on images of

different viewpoints and average matching times required to match one query image respectively.

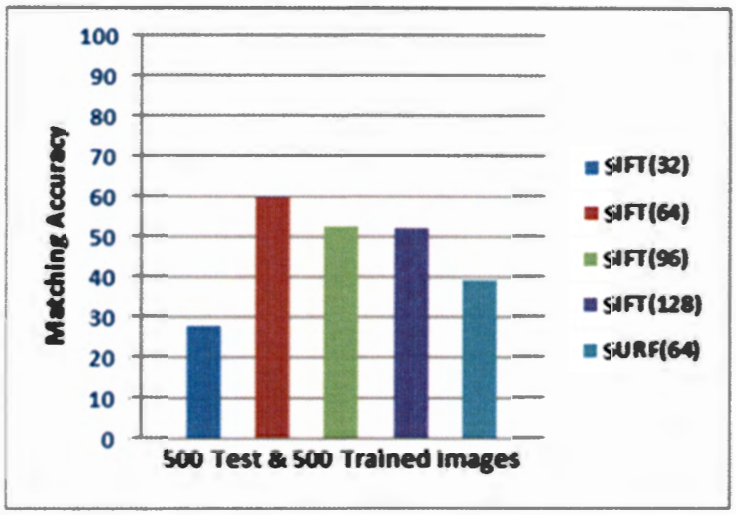


Figure 5.5: SIFT vs. SURF Performance [*75*]

Algorithm	**128D SIFT**	**96D SIFT**	**64D SIFT**	**32D SIFT**	**SURF**
Time (seconds)	59	33	18	11	80

Table 5.4: Average Matching Time [*75*]

In [*76*], Suaib et al. did a performance evaluation of the SIFT and SURF feature detectors and feature matching using a sample of the Amsterdam Hague dataset with a resolution of 640 × 480 pixels running at 30 *Hz*. Figure 5.6 shows the results from their experiments on a comparative performance evaluation of the SIFT and SURF techniques with regards to rate of matched points.

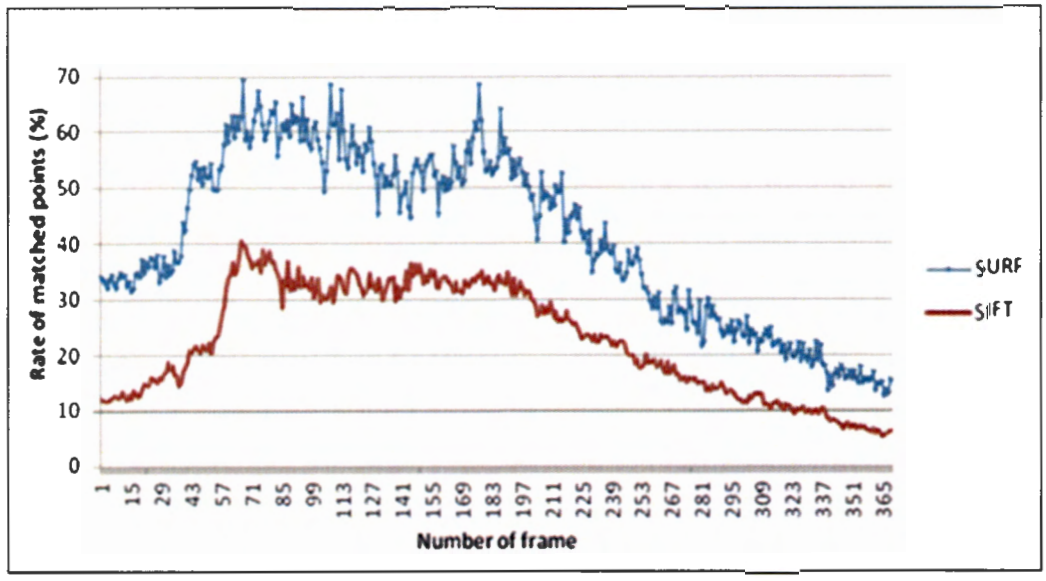


Figure 5.6: Comparison of SIFT and SURF Performance [*76*]

Basing on the results of the experiments from both publications [*75*] and [*76*], it can be inferred that the SIFT technique performed better on feature points matching and average time (in seconds) required to match a query as compared to the SURF.

5.1.5 The Hough Transform

Ramsoful et al. in [*39*] carried out some experiments aimed at evaluating the performance of a certain biometric security system that used the generalized Hough Transform technique. One important thing to note is that the Hough Transform that was used in the biometric security system applied the Mahalanobis distance to calculate its parameter values.

In order to test the performance of the biometric security system, 500 images obtained from 100 individuals were used and each individual had 5 instances which were taken at different intervals. The evaluation looked at the technique's response in the following performance measures.

1. False Acceptance Rate (FAR) which is the probability that an unauthorized person gets access to the system.
2. False Rejection Rate (FRR) which is the probability that an unauthorized person is rejected.

3. Equal Error Rate (EER) which is the point where the FAR is equal to FRR.

Table 5.5 shows the FAR and FRR for 10, 20, 30, 40, 50,100, 200 and 400 images tested respectively.

Number of Images	FAR (%)	FRR (%)
10	0.2000	0.3000
20	0.1000	0.0500
30	0.0667	0.0667
40	0.0750	0.0250
50	0.0200	0.0200
100	0.0100	0.0100
200	0.0000	0.0050
400	0.0000	0.0025

Table 5.5: Performance Measures for Hough Transform [*39*]

To show the EER which was defined as the points where the FAR and the FRR have equal values. Figure 5.7 presents a graph, the Receiver Operating Curve that plots the FAR and FRR points thereby showing the points where the two measures have the same values (EER).

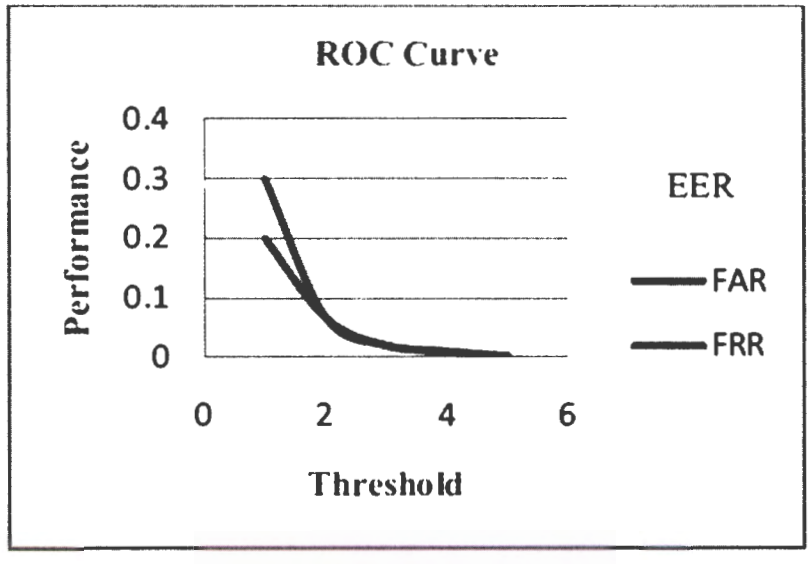


Figure 5.7: ROC Curve for Hough Transform [*39*]

5.2 Performance Evaluation Metrics

In this section, we introduce two widely used performance metrics for evaluating algorithms, the Fitness Value (FV) and the Efficiency Measure (EM) method implemented using the Big-Oh notation method.

5.2.1 Fitness Value Method

The fitness value method was applied using experiments that implement the GA. Firstly; we represented each design solution in genotype then transformed it into a chromosome (a string of binary numbers) and perform required computations by way of simulation runs. The idea behind these simulations was to discard the n^{th} worst design solution(s), known as weak/ unfit solutions then breed *'n'* new solutions from the best design solutions known as good/ fit solutions in each generation.

To accomplish that, we adopted a mechanism to award a figure of merit to each design solution as an indication of how close the solution is to the expected specification (optimum solution value). Equations 43 and 44 helped us to generate the fitness value that we used to qualify every solution generated in each simulation cycle (generation).

5.2.2 Efficiency Measure

EM requires that we look at two important aspects of the algorithm, which are *complexity* and *performance.* Looking at the two, we found that with performance, we are mainly concerned about how much *time, memory or disk space* is in use when we run an algorithm.

This translates to the type of instructions used in the algorithm, the data structures used to build the algorithm and their effects on runtime (execution duration) and space (memory used). From the discussion above, performance requires that we execute the algorithm on the computer and noting down the time and space values obtained, however from our reviews we found that efficiencies of different algorithms cannot be compared by running the algorithms on computers, as run-time is system and language dependent [*77*]. With this in mind, we saw

it logical to estimate the performance of the algorithms (GenApp, GF & MGF) by analysing the complexity of each algorithm, as the complexity of the algorithm has a direct effect on its performance. Hence, its follows that an algorithm with fewer complexities performs better.

5.2.2.1 The Big Oh Notation

In order to test the complexity of the algorithms we used the asymptotic analysis technique known as the *"Big Oh"* Notation, denoted by "O(n)". The rational being that the technique enables us to analyse the scaling of the resource requirements of the algorithms as the input data grows. We can achieve this by approximating a cost function that allows us to predict growth rate and express the general behaviour of the algorithm as the input data size grows larger respectively.

Computing the worst and best cases of the algorithms helps us to qualify them given that the worst case guarantees that the performance of the algorithm will be at least as good as the analysis indicates while the best case gives the best statistical estimate of the actual performance. In our review, we are mostly interested in the worst-case scenario which gives us the maximum number of operations that might be performed for a given problem size which helps us to evaluate the efficiency of the algorithms with respect to time and space.

Figure 5.8 shows some typical growth rate functions that we used to approximate growth rates of the algorithms that are as follows;

- $O(n)$ – Linear execution
- $O(n^2)$ – Quadratic execution
- $O(n^k)$ – Polynomial execution
- $O(2^n)$ – Exponential execution
- $O(n\log n)$ – Logarithmic execution

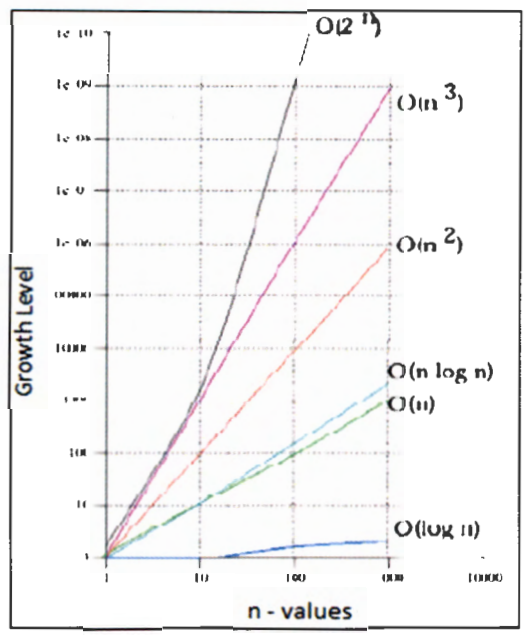


Figure 5.8: The Big Oh notation growth rate functions graph [*77*] .

5.3 Analysis and Evaluation of Algorithms.

In this section, we present the evaluations and results obtained from the two performance metrics discussed in sections 5.2.1 and 5.2.2 that is, the EM and the FV aimed at testing the performance of the algorithms. Bashir et al, in [*78*] identified a good algorithm as one that has fast response time as the algorithm's response time translates to the performance of the system that the algorithm is applied.

5.3.1 Efficiency Test

The efficiency measures of the GenApp, GF and MGF were evaluated by initially identifying the critical sections of the algorithm and analysing their behaviour separately using the Big O notation (analysis 1-7), followed by classification of critical regions based on their Growth Rate Functions (GRF) as illustrated in Figure 5.8. In our evaluations, we considered n and m as the current population and the total number of generations respectively.

5.3.1.1 Efficiency Test for GenApp

Analysis 1:

1. Initialize ();

$O(n)$ where n is the population size → Linear Execution

2. Compute Extrema (H, S) ID Strategy (equations 22-27);
3. Generate (P_c), where $(S \leq P_c \leq H)$;
4. Compute Fitness (F(s) (equation 43)

$O(n)$ where n is the population size → Linear Execution

5. Evaluate (P_c); *// AR ranking method*
6. If found Best Value, Output(Res), Stop
7. Else
8. *Res =BestOf* (P_c);

The algorithm steps in analysis 1 that is steps 2-4 requires that the algorithm loop through all the elements of the initial population P_c in a sequential manner to determine the highest and smallest values that will act as extrema. As a result, we can classify those steps as *Linear Execution* $O(n)$, while steps 5-8 loop through the P_c (current population) with some condition to select the best elements from the current population that will form part of the next generation

Analysis 2:

$O(n^2)$ where n is the population size → Quadratic Execution

9. Repeat_Loop1 until Stop (Criterion)
10. $P_n = 0$;
11. Repeat_Loop2 (P_c)/2 times *// UX ratio of 0.5*
12. Select $P_1 \in P_c, P_2 \in P_c$;
13. Crossover (P_1, P_2, C_1, C_2);
14. $P_n = P_n \cup \{C_1, C_2\}$;

15. End Repeat_Loop2;
16. $\forall p \in P_n$, Compute Fitness (F(s));
17. P_c= BestOf (P_c) ∪ BestOf (P_n);
18. Evaluate (P_c);
19. $\forall p \in P_c$, Possible Mutate (p) *//RMHC method with a probability of 0.01*

$O(n)$ Where n is the population size → Linear Execution

20. Evaluate(P_c)
21. Res = BestOf($P_c \cup \{Res\}$);
22. End Repeat_Loop1
23. Output (Res);

Analysis 3:

---

10. $P_n = 0$;

$O(n\log_2 n)$ Where n is the population size → Logarithmic Execution

11. Repeat_Loop2 (P_c)/2 times *// UX ratio of 0.5*
12. Select $P_1 \in P_c, P_2 \in P_c$;
13. Crossover (P_1, P_2, C_1, C_2);
14. $P_n = P_n \cup \{C_1, C_2\}$;
15. End Repeat_Loop2;

Analysis 2 has steps 10-22 iterated n times while steps 12-14 iterated m times. This means that the critical section in analysis 2 is looped $\left(n \times \frac{m}{2}\right)$ times, of which when $m = 2n$, it converges to n^2, hence classified as *Quadratic Execution* $O(n^2)$. Again, step 21 performs some evaluations that will require the algorithm to branch depending on the outcome; hence, we approximate it to *Linear Execution* $O(n)$. In analysis 3, we see iterations on steps 12-14, $\frac{m}{2}$ times, of which we can safely approximate it to a *Binary or Logarithm execution* $O(n \log_2 n)$.

5.3.1.2 Efficiency Test for GF

Analysis 4:

1. At each given point (x, y); compute the θ-ID values for the set $T = \{\theta_1, \dots, \theta_n\}$ as follows;

   $O(n)$ where n is the population size → Linear Execution

   $\theta\text{-}ID_{x,y}^{T} = \left\{\theta\text{-}ID_{x,y}^{\theta_1}, \dots, \theta\text{-}ID_{x,y}^{\theta_n}\right\}$

2. For each set compute the parameters of the highest local maximum and smallest local minimum as follows;
3. **$O(n)$ where n is the population size → Linear Execution**

   HLM: $(\sigma_i^{max}, \gamma_i^{max}) = \arg\max_{\sigma,\gamma} \theta\text{-}ID_{x,y}^{T}$

   SLM: $(\sigma_i^{min}, \gamma_i^{min}) = \arg\min_{\sigma,\gamma} \theta\text{-}ID_{x,y}^{T}$

4. Then the set of chosen GF parameter will be:
5. **$O(n)$ where n is the population size → Linear Execution**

$$P_{\theta}^{min} = \{(\sigma_i^{min}, \gamma_i^{min}, \theta_i), \dots, (\sigma_n^{min}, \gamma_n^{min}, \theta_n)\}$$
$$P_{\theta}^{max} = \{(\sigma_i^{max}, \gamma_i^{max}, \theta_i), \dots, (\sigma_n^{max}, \gamma_n^{max}, \theta_n)\}$$

Analysis 5:

6. Represent the feature vector as;

   $O(n^2)$ where n is the population size → Quadratic Execution

   $v_{(x,y)} = (v_{(x,y)}^{1}, \dots, v_{(x,y)}^{i}, \dots, v_{(x,y)}^{m})^{T}$

In analysis 4, steps 1, 3 and 5 execute sequentially looping across all the elements in the population, hence we classified it as *Linear Execution* $O(n)$. Analysis 5 step 7, the critical section is looped (m × i) times of which when $i = m$ and $m = n$ it converges to n^2 , hence classified as *Quadratic Execution* $O(n^2)$.

Analysis 5:

6. Represent the feature vector as;

   $O(n^2)$ where n is the population size → **Quadratic Execution**

   $v_{(x,y)} = (v^1_{(x,y)}, \ldots, v^i_{(x,y)}, \ldots, v^m_{(x,y)})^T$

Analysis 5 step 6, the critical section is looped (m × i) times of which when $i = m$ and $m = n$ it converges to n^2 , hence classified as *Quadratic Execution* $O(n^2)$.

5.3.1.3 Efficiency Test for MGF

Analysis 6:

$O(n)$ Where n is the population size → **Linear Execution**

1. Divide the input fingerprint image into blocks of size $W \times W$

$O(n^2)$ Where n is the population size → **Quadratic Execution**

2. Compute the gradients G_x and G_y at each pixel (x, y) in each block.

Analysis 7:

$O(n^3)$ Where n is the population size → **Polynomial Execution**

3. Estimate the local orientation of each block using the following formula:

$$\theta(i,j) = \frac{1}{2}\tan^{-1}\left(\frac{\sum_{u=i-w/2}^{i+w/2}\sum_{v=j-w/2}^{j+w/2} 2G_x(u,v)G_y(u,v)}{\sum_{u=i-w/2}^{i+w/2}\sum_{v=j-w/2}^{j+w/2}\left(G_x^2(u,v)-G_y^2(u,v)\right)}\right)$$

In analysis 6, Step 1, loops through the size of the population creating blocks of similar size $(W \times W)$ in a sequential manner. As a result, we classified it as *Linear Execution* $O(n)$. While step 2 is a bit different from step 1 in the sense that each time we compute gradient we first determine the values of x then y, hence the critical section loops $(p \times q)$ times where p

and q are the number of x and y elements in the population. In situations where $q = p$, the critical section converges to p^2, hence classified as *Quadratic Execution* $O(n^2)$.

In Analysis 7, the critical region executes the same way as in analysis 6 step 2 but in addition to that, it has some procedure calls $G_x(u, v)$ $G_y(u, v)$ embedded. As a result, the critical region has a complexity of $O(n^2)$ and $O(n)$ executed concurrently, of which when combined, they converge to $O(n^3)$ *polynomial.*

5.3.2 Algorithm Comparison

In order to compare the efficiency of the algorithms discussed, we looked at the results obtained from the evaluations on the complexities of the algorithms (analysis 1- 7). To achieve that, we assigned weights to each growth rate function based on figure 5.8, that is, as a percentage of the growth level. Selecting an arbitrary generation value (n) defined for all growth rate functions followed, of which in our case we considered $n = 100$ (100 generations).

Using the Big Oh, an algorithm that is $O(n^3)$ is said to be "harder" to compute than one that is $O(n^2)$ because the former will require, in general, more operations to complete than the latter, regardless of the speed at which these operations are performed [*79*].

Table 5.6 presents a combination of weighted growth rate functions as illustrated in analysis 1-7. Considering for example, the GenApp has three instances of *Linear Execution* $O(n)$ evaluated to a weighted value of 0.6, one instance of *Quadratic Execution* $O(n^2)$ that evaluates to a weighted value 0.4 and another instance of *Logarithm execution* $O(n \log_2 n)$ that evaluates to a weighted value 0.23. Since these instances happen in a sequential manner as the algorithm executes, we summed the weighted values to get a total complexity value of 1.23 as shown in table 5.6.

Algorithm comparison based on complexity						
GR Functions	$O(n)$	$O(n^2)$	$O(n^3)$	$O(\log_2 n)$	$O(n \log_2 n)$	Total
Weights	0.2	0.4	0.6	0.04	0.23	
Algorithms						
GenApp	3	1	0	0	1	
	0.6	0.4	0	0	0.23	1.23
GF	3	0	0	0	0	
	0.6	0	0	0	0	0.6
MGF	1	1	1	0	0	
	0.2	0.4	0.6	0	0	1.2

Table 5.6: Algorithm Complexity Results Based on Growth Rate Functions Weights.

5.3.2.1 Worst Case Analysis

Apart from finding the complexity of the algorithms, we felt it was equally important to check the relation of the number of operations to the problem size. This check helped us to determine when the algorithms will perform *worst* for any given input values.

The worst-case analysis gives an estimated upper bound on the resources required by the algorithm, which is the longest running time by an algorithm, given a variable input data of size *n*. Considering the GenApp algorithm, its worst-case scenario could be finding extrema values (HLM & SLM) if the initial population elements aren't grouped in some particular order. In this case, the algorithm has to first order the elements into some particular order (ascending/ descending) by applying a sort algorithm, say bubble sort, which requires the algorithm to iterate through all the elements in the initial population. Therefore, in such a case the time needed for finding extrema values is proportional to the number of elements in the initial population $O(n)$.

The GF and the MGF share similar worst-case scenarios but they both suffer a major drawback in that they do not screen the initial population, meaning to say if the initial population is mainly composed of weak elements, the algorithms start from a very weak position hence will have to iterate through several generations in order to improve their results. From analysis 4, 5 & 3, 7 in sections 5.1.2 and 5.1.3, we classified the GF and the MGF with growth rate functions $O(n)$, $O(n^2)$ and $O(n)$, $O(n^2)$, $O(n^3)$ respectively. The algorithms show that these functions are iterated in each generation cycle throughout all the predefined generations. An increase in input size of the algorithms result in a more rapid growth in resources (time and space) making them less efficient as indicated in figure 5.8.

5.3.3 Experiments and Results

In this part our experiments to test the performance of the GenApp were performed in form of simulation runs using genAID with the standard Microsoft Excel sheets being the source and repository of the simulation runs.

5.3.3.1 Fitness Value Test

In order to measure the fitness of the candidate parameter values that were selected by FET on each generation, we performed three tests. These tests were aimed at evaluating how the GenApp eventually obtain an optimum parameter value by applying a genetic evolution model. In this case, we only managed to perform fitness value tests on the GenApp owing to the heterogeneity nature of some of the previously discussed techniques namely the MGF and the GF. However, this had little impact on our study given that our focus was mainly on checking the rate at which the *GenApp* optimised its input values.

The data we used for our experiments was simulated data that we got arbitrarily from using a random number generator (RNG). In order to control the output from our generator, we had to apply algorithm 4 adopted from [***80***] in such a way that the values generated follow a certain criterion that represented the initial candidate values as applied to the FET parameters in real situation. Our generator produced candidate values in phenotype that is floating point numbers (5 decimal points) which we in turn converted to genotype as chromosomes commonly known as binary digits of 16 bits length.

This conversion (floating point to binary integers) enabled us to apply basic genetic operators, which are selection, crossover, and mutation that we used to compute the chromosomes over several generations in our simulations as way to gradually improve their fitness values [***81***]. The fitness value of each candidate chromosome was calculated relative to its adjacent candidates using equation 43 in such a way that the chromosomes had an opportunity to compete against each other for survival.

Algorithm 4: Controlling the RNG Output

1. Input seed
2. $Q = \{seed\}$
3. While $Q.\text{NextElement} \neq \{\}$
   (a) For each x value in $Q.\text{NextElement}$, if $P(x, Q) = \text{true}$, then add x to Q
   (b) End While
4. Return Q

In algorithm 4, Q represent the simulated initial point on the RoI in real situation, while x is any point on the RoI and $seed$ being an arbitrary integer value used to initialize the RNG.

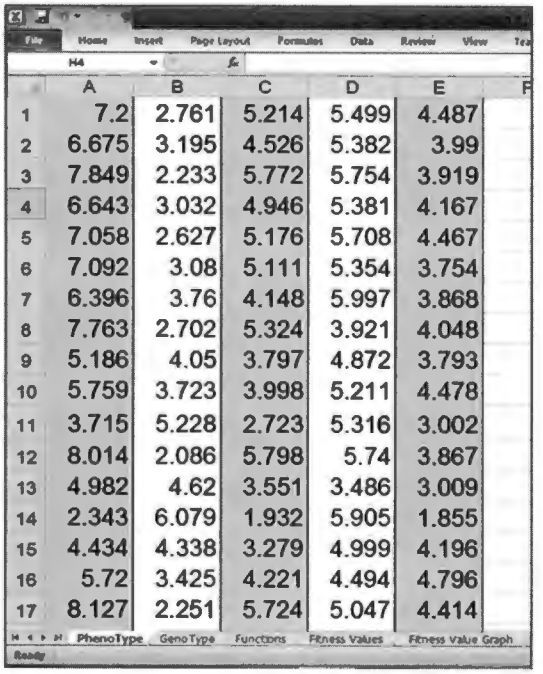
5.3.3.2 Simulation Plan

Testing the performance of the GenApp required us to draw a plan on the parameters and their corresponding values to be used on the simulator. Five simulation runs were performed with each simulation having its own carefully selected set of parameters and their corresponding values. This was meant to test how the new approach reacts in different scenarios provided thereby making recommendations on the scenarios the approach gives better results. Table 5.7 summarises the implemented test plan.

Simulations 4 and 5 had similar parameter values except for the crossover probability. The idea was to check how the GenApp reacts to producing offspring for the next generation using different weights on the mating parents during crossover operation. Figures 5.9 and 5.10 illustrate sample data represented in phenotype and genotype respectively that we generated for use in our simulations. The headings A to E on Figure 5.9 represent the data obtained from the random generator that was used as initial population for the simulations runs 1 to 5.

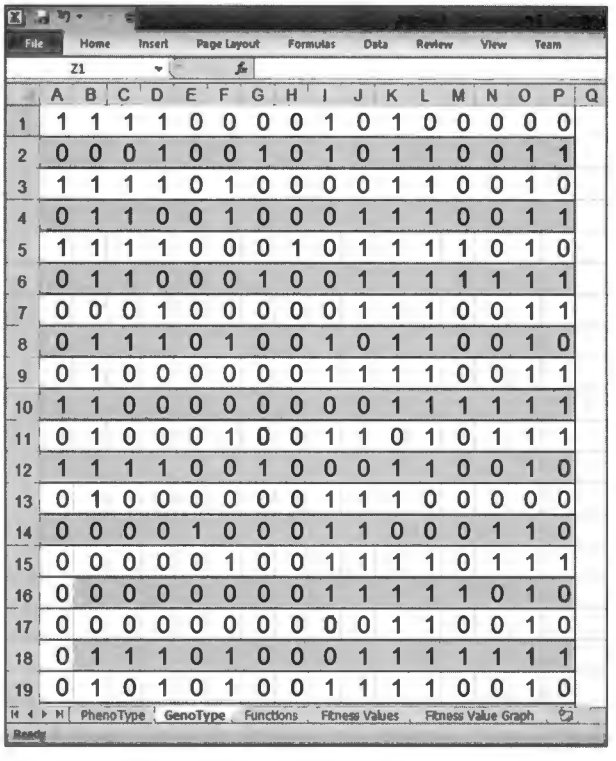
genAID Parameters	Simulations				
	1	2	3	4	5
Population Size	20	80	185	305	305
No. of Generations	12	50	120	200	200
Crossover Probability	0.5	0.5	0.5	0.5	0.25
Mutation Probability	0.01	0.01	0.01	0.01	0.01

Table 5.7: genAID Simulation Parameters and their Corresponding Values



	A	B	C	D	E
1	7.2	2.761	5.214	5.499	4.487
2	6.675	3.195	4.526	5.382	3.99
3	7.849	2.233	5.772	5.754	3.919
4	6.643	3.032	4.946	5.381	4.167
5	7.058	2.627	5.176	5.708	4.467
6	7.092	3.08	5.111	5.354	3.754
7	6.396	3.76	4.148	5.997	3.868
8	7.763	2.702	5.324	3.921	4.048
9	5.186	4.05	3.797	4.872	3.793
10	5.759	3.723	3.998	5.211	4.478
11	3.715	5.228	2.723	5.316	3.002
12	8.014	2.086	5.798	5.74	3.867
13	4.982	4.62	3.551	3.486	3.009
14	2.343	6.079	1.932	5.905	1.855
15	4.434	4.338	3.279	4.999	4.196
16	5.72	3.425	4.221	4.494	4.796
17	8.127	2.251	5.724	5.047	4.414

Figure 5.9: A Sample of the Simulated Random Number Generator Output in Floating Point Representation (Phenotype)



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	1	1	1	1	0	0	0	0	1	0	1	0	0	0	0	0
2	0	0	0	1	0	0	1	0	1	0	1	1	0	0	1	1
3	1	1	1	1	0	1	0	0	0	0	1	1	0	0	1	0
4	0	1	1	0	0	1	0	0	0	1	1	1	0	0	1	1
5	1	1	1	1	0	0	0	1	0	1	1	1	1	0	1	0
6	0	1	1	0	0	0	1	0	0	1	1	1	1	1	1	1
7	0	0	0	1	0	0	0	0	0	1	1	1	0	0	1	1
8	0	1	1	1	0	1	0	0	1	0	1	1	0	0	1	0
9	0	1	0	0	0	0	0	0	1	1	1	1	0	0	1	1
10	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1
11	0	1	0	0	0	1	0	0	1	1	0	1	0	1	1	1
12	1	1	1	1	0	0	1	0	0	0	1	1	0	0	1	0
13	0	1	0	0	0	0	0	0	1	1	1	0	0	0	0	0
14	0	0	0	0	1	0	0	0	1	1	0	0	0	1	1	0
15	0	0	0	0	0	1	0	0	1	1	1	1	0	1	1	1
16	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	0
17	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0
18	0	1	1	1	0	1	0	0	0	1	1	1	1	1	1	1
19	0	1	0	1	0	1	0	0	1	1	1	1	0	0	1	0

Figure 5.10: A Sample of the Random Number Generator Representation in Binary Digits (Genotype)

5.3.3.3 Simulation Results

Figures 5.11 to 5.15 show the results of simulations runs 1 to 5 respectively that we conducted using sample data and parameter values presented in Figures 5.9, 5.10 and Table 5.6. In each simulation run, the fitness values are presented graphically with two lines upper and lower representing the maximum and average fitness values of the candidates in floating point on each generation created respectively.

In all the five experiments a relatively consistent data as shown in Figures 5.9 and 5.10 was used, however we had to consider only the average and the maximum fitness values as the minimum fitness values have a tendency of trapping the algorithm at local minimum. Mutation probability was kept at a very low value of 0.01 to avoid major deviations in fitness values from one generation to the other which have an effect of prolonging the yielding of the algorithm. The crossover value remained unchanged at 0.5 in the first four simulations runs to allow a 50 percent contribution from each parent value to the next generation offspring. While simulation run 5 varied the parentage contribution of the two parents to 25 and 75.

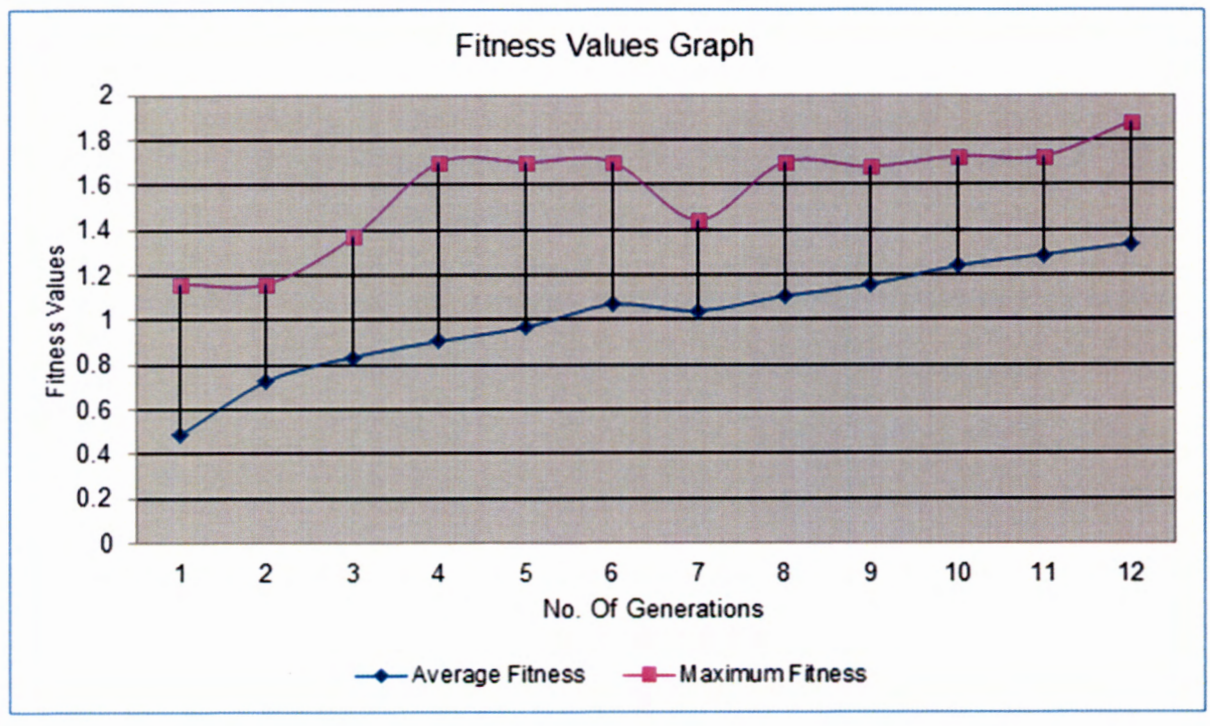


Figure 5.11: Simulation Run 1 Result from 12 Generations on Population Size of 12 and Mutation & Crossover Probabilities of 0.01 & 0.5 Respectively

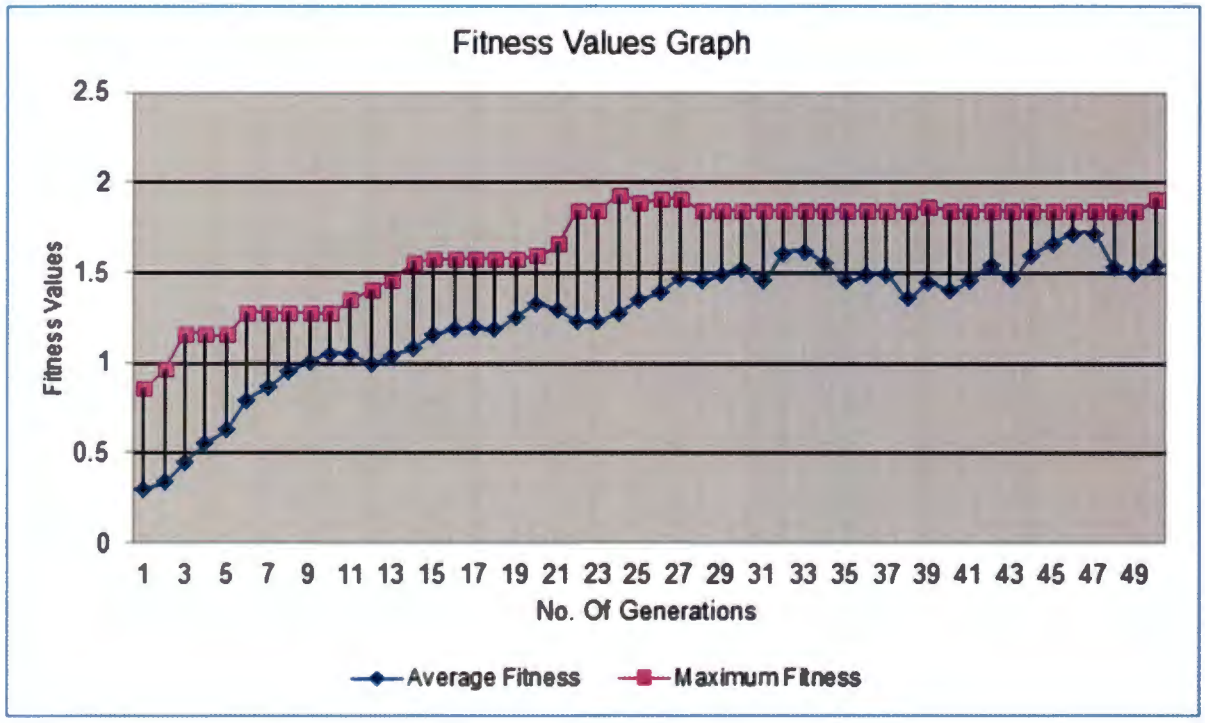


Figure 5.12: Simulation Run 2. (50 Generations on Population Size of 80 and Mutation & Crossover Probabilities of 0.01 & 0.5 Respectively)

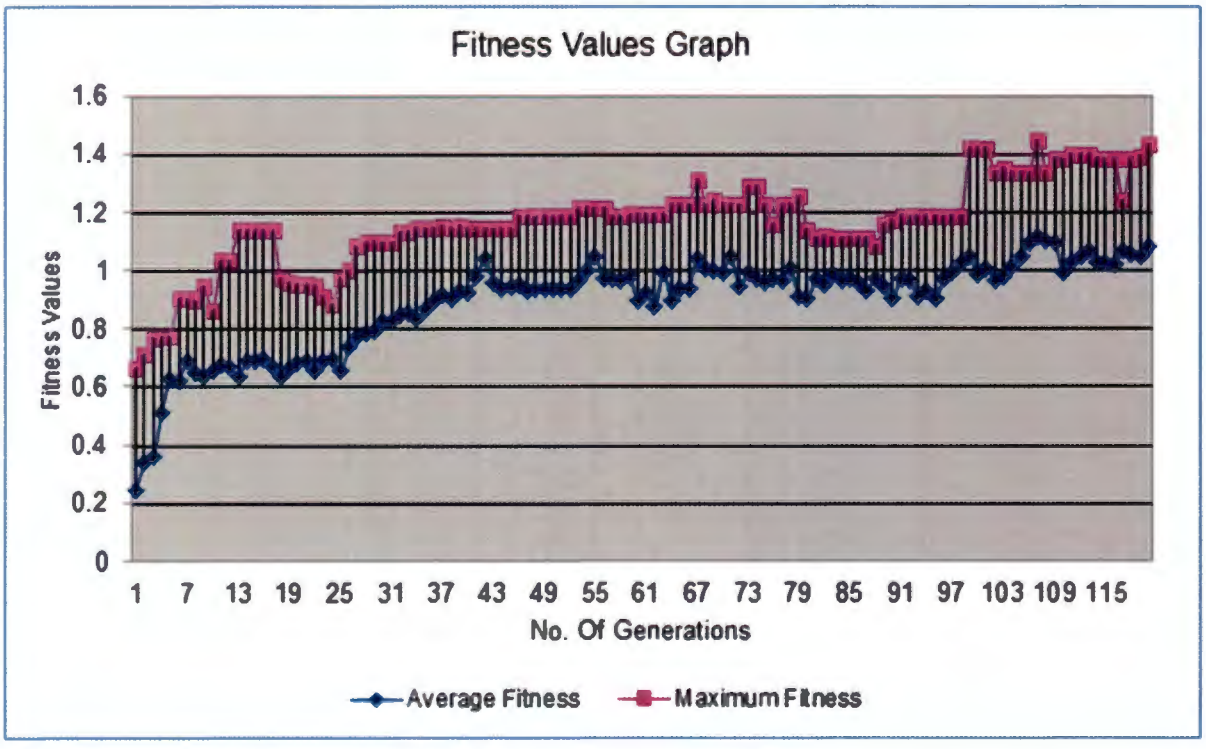


Figure 5.13: Simulation Run 3. (120 Generations on Population Size of 185 and Mutation & Crossover Probabilities of 0.01 & 0.5 Respectively)

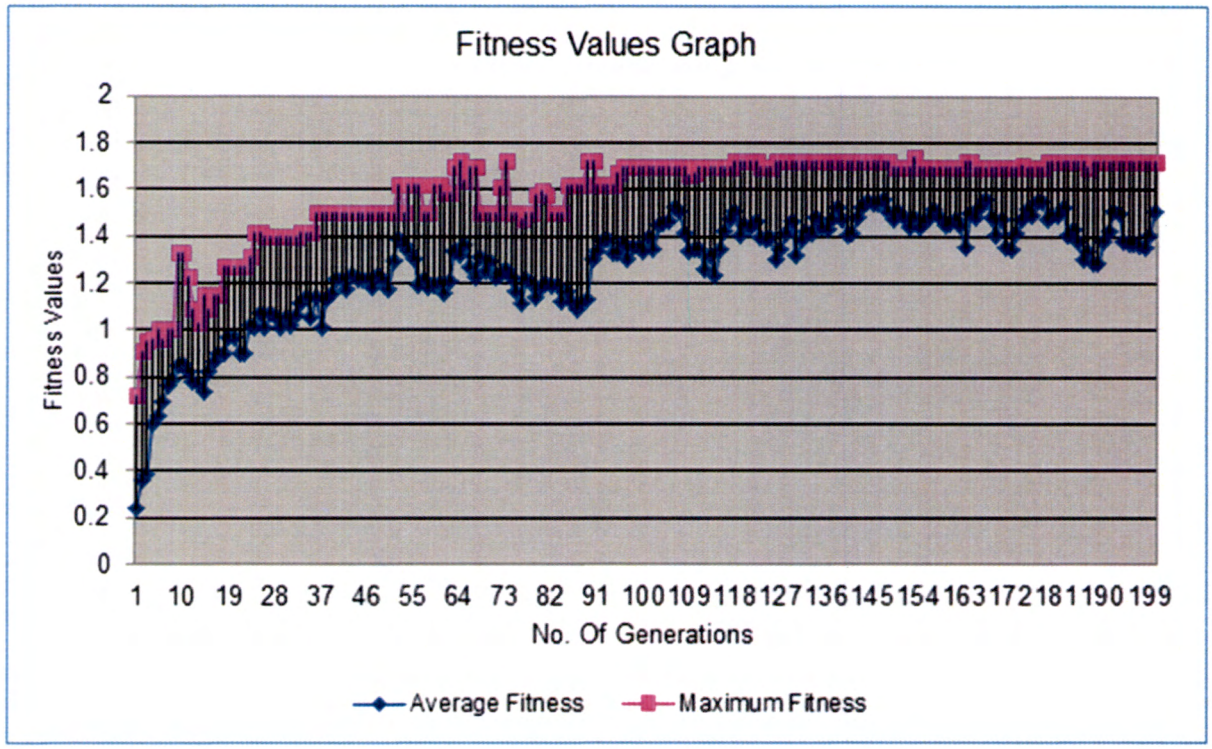


Figure 5.14: Simulation Run 4. (200 Generations on Population Size of 284 and Mutation & Crossover Probabilities of 0.01 & 0.5 Respectively)

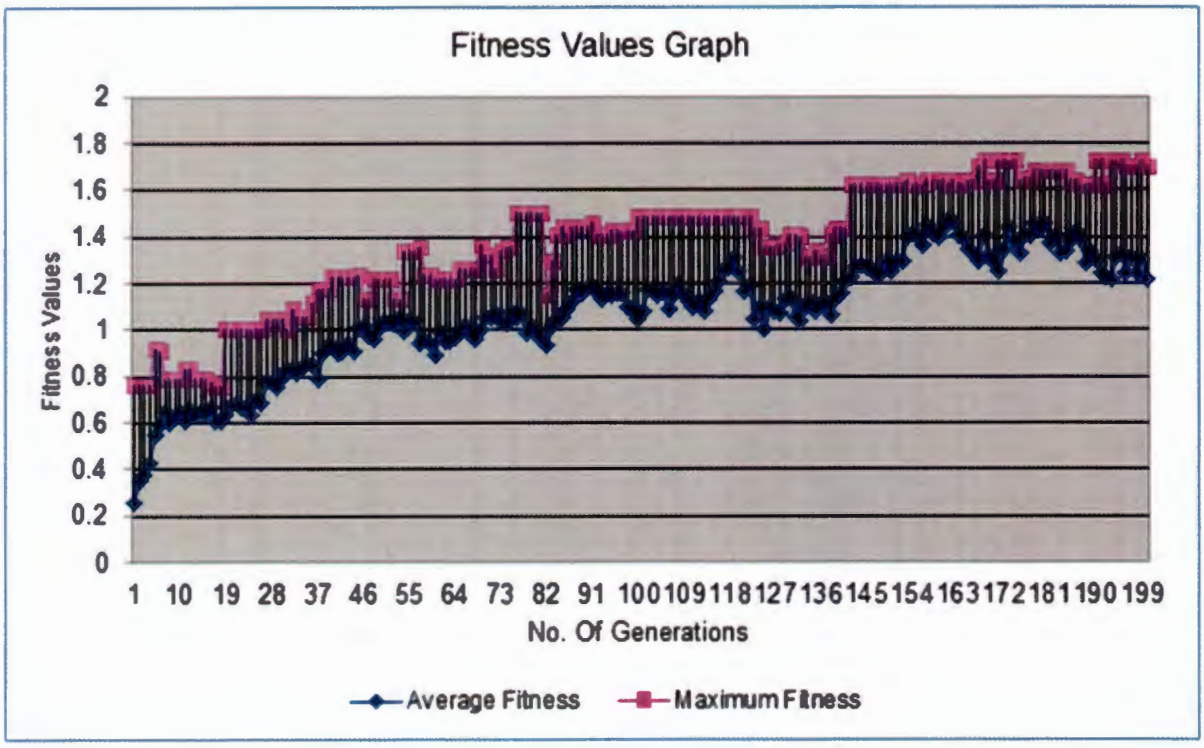


Figure 5.15: Simulation Run 5. (200 Generations on Population Size of 284 and Mutation & Crossover Probabilities of 0.01 & 0.25 Respectively)

Table 5.8 summarizes the results from the 5 simulations performed by giving the maximum fitness value achieved and average fitness value obtained in all the generation in a performed simulation run. Normalised point (NP) is the generation at which the fluctuations of the fitness value of the algorithm started to normalise while *% NP to G* is the percentage number of the generations required to normalise the algorithm's fitness value, calculated as $(\frac{NP}{G} \times 100)$.

Max. Fitness Value	1.5	1.9	1.42	1.78	1.78
Avg. Fitness Value	1.39	1.5	1.1	1.42	1.23
Total No. of Generations (G)	12	50	120	200	200
Population	12	80	185	284	284
Normalised Point (NP)	8	21	65	91	98
% NP to G	66.7	26.2	35.1	32.0	34.5

Table 5.8: Simulation Runs Results Summary

In all our simulation runs as indicated in Figures 5.11- 5.15, we witnessed a similar tendencies whereby the fitness value graph sharply rise in the initial stages of the generations then slowly becoming stable as the generations lapses. For example, in simulation run 4, the fitness value made a sharp rise between generations 1 and 55, reaching the maximum fitness value at 56 before it started to fluctuate between generations 64 and 83 before it normalized to 1.7. Normally the fluctuations experienced after reaching the maximum fitness such as the ones between generations 64 and 83 are a result of the 0.01 mutation value.

It can be observed as well from the results that the GenApp managed to optimize the fitness value at a much faster rate as indicated by the (% NG to G) in Table 12. Considering simulations 2-5, that is Figures 52-55, the average percentage of the generations required to obtain a maximum fitness value is approximately 31.95. This could be credited to the algorithm's ability to screen weaker candidates that is those prospective solutions with values that aren't within acceptable range from the initial population by using extrema (HLM & SLM). As a result, the algorithm needed not to iterate through many generations to find an optimum value for the parameters.

5.3.4 Discussion of Results

The performance and algorithm evaluations performed in sections 5.2 and 5.3 respectively measured the efficiency, fitness and the worst case scenarios of the GenApp in comparison to the existing algorithms.

5.3.4.1 Efficiency Test

The results presented in Table 5.6 illustrate the frequency at which each algorithm presented some significant degree of complexity based on the Big O Notation and its corresponding weighted GR function value. Comparing the resultant complexity value of the GenApp with the existing algorithms, that is the GF and the MGF. The GF posted an accumulated value of 0.6, followed by the MGF and the GenApp with values of 1.2 and 1.23 respectively.

The GenApp posted the highest complexity values of 0.63 and 0.03 above the existing algorithm owing to its initialization steps that sort the input data to enable it to screen the initial population. This improves the worst-case scenario of the GenApp algorithm hence could be seen as a trade-off in its overall performance as compared to other existing algorithms that fail to screen the initial population.

5.3.4.2 Fitness Tests

Table 5.7 shows the input values we picked at random for 5 different simulation runs. The results obtained in these simulations helped us to understand the behavior of the GenApp when presented with different population sizes ranging from as little as 20 to 305 initial input values. The research was mostly interested in observing the GenApp's consistency in optimizing the initial population; hence we did not compare it with the existing algorithms. Figures 5.11 to 5.15 show graphical representations of the fitness values obtained from the simulations with Table 5.8 showing a rather consistent average fitness on all the simulations. This gave us confidence in the algorithm's ability to consistently optimize the initial population without major deviations.

5.4 Chapter Summary

The chapter reviewed performance evaluation of different FET. The PCA performed better than the ICA and LDA on both metrics, the Recognition Rate and F-Score, while efficiency and robustness were the other two metrics that were considered for testing. Although the GenApp showed a higher complexity values as compared to the existing algorithms it still has a chance of presenting better results given that the complexity was affected mostly by its initial population screening phase which helps reduce the number of generations required to produce an optimized value.

The worst case scenarios for all the GF, MGF and GenApp algorithms were analysed and the results showed that similar tendencies on the GF and MGF in that they failed to screen the initial population. This result in the two algorithms starting from a weaker position in situations where the initial population has some weak elements. The GenApp's strength was in screening weaker elements in the initial population resulting in the algorithm starting the optimization process from a relatively strong position. However its major drawback was a situation where the elements in the initial population aren't ordered in some particular order (Ascending/ Descending), screening elements was a challenge.

Simulations on the fitness values using data in phenotype and genotype showed the GenApp managing to optimize the parameter values in most cases in first quarter of the generations specified. However, a change in the mutation percentage (simulation run 5) resulted in the algorithm getting the same fitness value but at an increased number of generations and this explains the reason why we kept it at a low 10 percentage.

Chapter 6: Summary, Conclusion and Future Work

6.1 Summary

Summarily, the research study presented in this thesis focused on developing the novel adaptive algorithm that we named the GenApp based on the adaptive principles of the biological genes. The development of the algorithm followed the contemporary algorithm development phases as outlined in figure 4.5. The most fascinating thing about the GenApp was the fusion of different concepts into the algorithm such as the information datagram concept that played an important role of screening the initial population and the genetic operators that selected and optimized the parameter values in an iterative manner through different generations.

The use of empirical experiments or data obtained from empirical experiments dominated in most researches we reviewed. Although acceptable results were obtained through this approach, the bias and short runs associated with them didn't give us much comfort in the conclusions and outputs obtained. This approach in some cases tends to distribute parameters to a narrow range which in most cases negatively affects the capability of the approach to discriminate the modeled object.

This study is quite important in that it contributed to the literature in RSI processing by providing an approach that can be used to improve parameter selection strategy for FET which has long been a research focus in the field of image processing. As a result of this study we should be able to among other things, increase detail and accuracy of images obtained through FET in RSI processing as well as proposing a more flexible parameter selection scheme.

In order to test the GenApp algorithm, two performance evaluation metrics, the fitness value test and the Big Oh notation were implemented separately. These two metrics aimed at evaluating the efficiency and the fitness of the algorithm respectively. Analysis 1-7 tested the efficiency of three algorithms including the GenApp with table 5.6 presenting the results obtained in a comparative manner. Figures 5.11 to 5.15 illustrate the results graphically

obtained from the simulations we did using randomly generated data to check the fitness of the GenApp. It is important to note that the average and maximum fitness was considered in our experiments in this research work. This decision was reached after discovering that considering minimum fitness values had a tendency of trapping the algorithm to a local minimum, which failed to produce desirable results for our experiments.

6.2 Conclusion

Conclusively, parameter selection plays a crucial role in the use of FET if we are to get better output images. Although a number of solutions could be thought of to address the issue of parameterization, we chose to implement genetic algorithms owing to their strength in optimization.

The research studies the characteristics of currently used FET in existing literature, performs some requirements elicitation for dynamic parameterization, develop an adaptive algorithm named the GenApp that implements genetic algorithms and finally perform an evaluation on efficiency and fitness of the developed algorithm using MatLab and GenAID as simulation tools.

In the research we provided some questions that helped us to focus on our goal. In response to these questions we modeled the adaptive principles of the biological genes into our algorithm some applying commonly used genetic operators namely, selection, crossover and mutation. The operators were applied to an initially selected population that contained supposedly a blurred region of interest.

Screening the elements in the initial population is quite important in this process, as it reduces the optimization task by discarding those elements that are not close to the required solution value. These elements require more generation cycles to be optimized thereby consuming more time. In this case the Average Ranks method was implemented in the selection to determine the potential candidates that would be passed to the crossover operator.

The elements that passed into crossover stage are propagated using a uniform mixing ratio of 0.5, forming new offspring that are optimized. The Random Mutation Hill Climbing method was applied to select a candidate element and compare it with its potential neighbors only if

improves its value. To achieve this, a mutation ration of 0.001 was considered to help the algorithm avoid major deviations from the value obtained during crossover process.

The research help us deviate from relying on empirical data as observed in the GF and MGF where some parameters such as σ_y were given fixed. This model is likely to fail when applied in an environment is not constant, hence the need for an algorithm or model that specifies parameter values adaptively.

To justify the goal of the thesis, simulations and experiments were performed in Chapters 4 and 5 to test the complexity, worst-case scenarios and fitness of the GenApp in comparison with other existing algorithms. The results were presented graphically and in table showing the performance all the algorithms. Although the GenApp would at times come last as in the test for complexity. The results obtained are within the same margins, which can be traded-off with its improved worst-case scenario. The good thing with the GenApp is that it managed to obtain an average fitness value within the first quarter of the generations and would consistently maintain it throughout the generations as illustrated in Table 5.8. This gives confidence in the algorithm's potential to optimize a given solution.

Based on the above findings we recommend the following;

(i) The use of algorithms that implement adaptive principles to improve the quality of the results obtained in RSI through FET.

(ii) The implementation of genetic algorithms for adaptive parameterization of FET in RSI.

6.3 Future Work

Feature extraction has many areas of specialization that can be looked into separately, for example, some algorithms have been designed to identify specific target objects while others focus more generally on say buildings or roads extractions. Although the extraction techniques for these different specializations can vary substantially, their strengths depend squarely on their ability to identify and apply optimum parameter values. Parameter optimization in feature extraction is a critical requirement that can never be overemphasized.

Basically, an effective feature extraction technique should be able improve an image with a corrupted or blurred region of interest.

There are many emerging disciplines in computer science that can be explored and applied to improve the existing approaches to feature extraction. Using evolutionary computing approaches, for example genetic algorithms as discussed in this research or neural networks could help enhance reasoning in algorithms thereby improving the overall performance of the whole feature extraction process.

In our research, we used simulated data generated using a controlled random generator. Although we managed to test the behaviour of the GenApp, we are still of the opinion it's important to test the algorithm under different natural conditions as generators tend to be biased towards some unrealistic tendencies.

Future work should now focus mainly on improving the GenApp in such a way that apart from adaptation it should apply some reason that mimic human behaviour. A hybrid model for example, which combines genetic algorithms and neural networks, might improve the algorithm by way of adding some reasoning capabilities.

As observed in all our experiments, the algorithm iterated through all the generations regardless of the fact that it had optimized the solution within the first quarter of the generations. Making the algorithm intelligent would be ideal to avoid resource wasting such as memory and time during execution. On the other hand, incorporating some intelligence into the algorithm helps it as well to notice the quality of the solution in such a way that if need be, it can proceed with the optimization process beyond the stipulated number of generations until it obtains the most optimum value that give a better resultant image from the extraction process.

References

[1] M Henrique and G Easson, *Feature Extraction from High-Resolution Remotely Sensed Imagery using Evolutionary Computation.* Mississippi, USA: Prof. Eisuke Kita, 2011.

[2] L. Liu, T. Jiang, Y. Fan, and J. Yang, "A Modified Gabor Filter Design Method For Fingerprint Image Enhancement," *Pattern Recognition Letters*, no. 24, pp. 1805 - 1817, January 2003.

[3] G. Joseph, *Fundamentals Of Remote Sensing*, 2nd ed. India: Universities Press Private Ltd., 2005.

[4] J. G. Daugman, "Uncertainty Relations For Resolution In Space, Spatial Frequency And Orientation Optimized by Two-Dimensional Visual Cortical Filters," *Optical Society Of America*, vol. 2, pp. 1160-1169, 1985.

[5] Y. Inoue, M. S. Moran, and E. M. Barnes, "Opportunities And Limitations For Image-Based Remote Sensing In Precision Crop Management.," *Remote Sensing environment*, vol. 61, pp. 319 - 346, January 1997.

[6] R. A. Schowengerdt, "Remote Sensing Models And Methods For Image Processing.," *Elsevier publication*, no. 3 2nd edition, 2006.

[8] N. Petkov and M. B. Wieling, "Gabor Filter For Image Processing And Computer Vision," *University of Groningen, Department of Computing Science and Intelligent Systems.*

[10] P Moreno, A Benardino, and J Santos-Victor, "Gabor Parameter Selection for Local Feature Detection," in *IBPRIA 2nd Iberian Conference on Pattern Recogniton and Image Image Analysis*, Portugal, 2005, pp167-172.

[11] V. Santhaseelan and V. K Asari, "An Adaptive Parameterization Method for SIFT based Video Stabilization," in *Applied Imagery Pattern Recognition Workshop, IEEE*, 2010, pp. 1-6.

[12] G. Kumar and P. K Bhatia, "A Detailed Review of Feature Extraction in Image Processing Systema.," in *International Conference on Advanced Computing & Communication Technologies.*, Rohtak, 2014, pp. 5-12.

[13] P. Kern, "Automated Feature Extraction Techniques: A History," in *MAPPS/ ASPRS 2006 Fall Conference*, San Antonio, Texas, 2006.

[14] V. Pali, S. Goswani, and L. P. Bhaiya, "An Extensive Survey on Feature Extraction

Techniques for Facial Image Processing," in *Sixth International Conference on Computational Intelligence and Communication Networks*, Bhopal, 2014, pp. 142-148.

[15] R. Cendrillon and B. C Lowel, "Real-Time Face Recognition using Eigenfaces," *Proceedings of the SPIE International Conference on Visual Communications and Image Processing.*, vol. 4067, pp. 269-276, 2000.

[16] I. J Cox, J. Ghosn, and P. N Yianilos, "Feature based face recognition using mixture-distance," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1996, pp. 209-216.

[17] J. Yiang, L. Liu, T. Jiang, and Y. Fay, "A modified Gabor Filter Design Method for Fingerprint Image Enhancement," *Pattern Recognition Letters*, no. 24, pp. 1805 - 1817, January 2003.

[18] J. Zhou, Z. Ji, L. Shen, Z. Zhu, and S. Chen, "PSO Based Memetic Algorithm for Face Recognition Gabor Filters Selection," in *Memetic Computing (MC). 2011 IEEE Workshop*, Paris, 2011, pp. 1-6.

[19] J Yang, L Liu, T Jiang, and Y Fan, "A modified Gabor Filter Design Method for Fingerprint Image Enhancement," *Pattern Recognition Letters*, no. 24, pp. 1805 - 1817, January 2003.

[20] S Fejes and F Vaida, "Simplified Adaptive Approach To Efficient Morphological Image Analysis," 2004.

[21] W. Lixia and J. Dalin, "The Application of Mthematical Morphology in Vehicle Identification Technology.," in *International Conference on Multimedia Technology (ICMT)*, HangZhou, 2011, pp. 801-805.

[22] E Saffor and A Salama, "Objective Evaluation of Mathematical Mophology Edge Detection on Computed Tomography (CT) Images," *International Journal of Medical Science and Engineering*, vol. 7, no. 9, pp. 631 - 635, 2013.

[23] V. Chatzis and I. Pitas, "A Generalized Fuzzy Mathematical Morphology and its Application in Robust 2D and 3D Object Representation.," *IEEE Transactions on Image Processing.*, vol. 9, no. 10, pp. 1798-1810, August 2000.

[24] B. Kaur and A. Garg, "Mathematical Morphological Edge Detection for Remote Sensing Images.," in *International Conference on Electronics Computer Technology (ECET), Volume 5*, Kanyakumari, 2011, pp. 324-327.

[25] Y. Wei and W. Tong, "Research on Edge Detection based on Mathematical Morpohology Algorithm. ," in *International Conference on Optoelectronics and Image*

Processing., Haiko, 2010, pp. 211-213.

[26] M. S Nixon and A. S Aguado, "Feature Extraction and Image Processing for Computer Vision," in *Mathematical Morphology*, Academic Press, Ed. London, United Kingdom: Elsevier Ltd., 2012, ch. 3, pp. 123-128.

[27] H. Wang, S. Yang, and W. Liao, "An Improved PCA Face Recognition Algorithm Based on the Discrete Wavelet Transform and the Support Vector Machines.," in *Computational Intelligence and Security Workshops (CISW)*, 2007.

[28] D. Jeyabharathi and A. Suruliandi, "Perfomance Analysis of Feature Extraction and Classification Techniques in CBIR.," in *International Conference on Circuits, Power and Computing Technologies (ICCPCT).*, 2013, pp. 1211 - 1214.

[29] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, January 2004.

[30] A. Mikhalev, E. J Hughes, and R. F Ormondroyd, "Comparison of Hough Transform and Particle Filter Methods of Passive Emitter Geolocation using Fusion of TDOA and AOA Data.," in *Information Fusion [FUSION], 2010 13th IEEE Conference* , Edinburgh, 2010, pp. 26-29.

[31] K. M Alexiev and L. V Bojilov, "A Hough Transform Track Initiation Algorithm for Multiple Passive Sensors.," in *Proceedings of the FUSION' 2000* , 2000, pp. TUB2/11-TUB216.

[32] H. T Chan and P. K Tam, "A New Hough Transform Form Based Position Estimation Algorithm.," in *IEEE Proceedings Australia and New Zealand Conference on Intelligent Information Systems*, 1994, pp. 140-144.

[33] W. S Yambor, "Analyzing PCA Based Face Recognition Algorithms: Eigenvector Selection and Distance Measures.," , 2000, pp. 1-4.

[34] J. Yun and R. Park, "Self-Calibration with Two Views Using the Scale Invariant Feature Transform. ," *Advances in Visual Computing.* , pp. 589-598, 2006.

[35] A. A Othman and H. R Tizhoosh, "Neural Image Thresholding with SIFT-Controlled Gabor Features," in *International Joint Conference on Neural Networks*, San Jose, California, 2011, pp. 2106-2112.

[36] W. Guang-hui et al., "An Algorithm of Parameters Adaptive Scale-Invariant Feature fpr High Precision Matching of Multi-source Remote Sensing Image ," in *Urban Remote Sensing Joint Venture2009, Proceedings to IEEE*, Shanghai, 2009, pp. 1-7.

[37] R. Li, M. A Henson, and M. J Kurtz, "Selection of Model Parameters for Off-Line Parameter Estimation.," *IEEE Transactions on Control Systems Technology*, vol. 12, no. 3, pp. 402-412, May 2004.

[38] W. Haihui, W. Yanli, Z. Tongzhou, W. Miao, and W. Mingpeng, "Images Segmentation Method on Comparison of Feature Extraction Techniques.," in *Intelligent System and Application (ISA), 2010 2nd International Workshop, IEEE*, Wuhan, 2010, pp. 1-4.

[39] P. Ramsoful and M. H Khan, "Feature Extraction Techniques for Dorsal Hand Vein Pattern. ," in *Innovative Computing Technology (INTECH), 3rd International Conference, IEEE*, London, 2013, pp. 49-53.

[40] F. Lin, C. Kao, and C. Hsu, "Applying the Genetic Approach to Simulated Annealing in Solving Some NP-Hard Problems," *IEEE Transactions on Systems Man. and Cybernetics*, vol. 23, no. 6, pp. 1752-1767, November 1993.

[41] N. Du, H. Peng, and W. Zhang, "Application of Modified Genetic Algorithm in Feature Extraction of the Unstructured Data.," in *International Conference on Advanced Computer Control*, 2008, pp. 124-128.

[42] N. N Rao, S. Srikanth, G. Vinay, and P. B Guru, "Genetic Algorithms-Assisted Feature Extraction and Selection for Global Motion Estimation," , 201.

[43] Pegah Hosseini, F. Almasganj, and M. R Darabad, "Pathological Voice Classification using Local Discriminant Basis and Genetic Algorithm," in *Mediterranean Conference on Control and Automation Congress Centre.*, Ajaccio, France, 2008, pp. 872-876.

[44] D. J Feng, S. Wijesoma, and A. P Shacklock, "Genetic Algorithmic Filter Approach to Mobile Robot Simultaneous Localization and Mapping," , 2006.

[45] L. Trujillo and G. Olague, "Synthesis of Interest Point Detectors Through Genetic Programming.," *ACM Press (GECCO 2006).* , vol. 1, pp. 887-894, July 2006.

[46] L. Fuyan, C. Chouyong, and L. Shaoyi, "An Improved Genetic Approach," in *International Conference on Neural Networks and Brain (ICNN&B), IEEE* , Beijing, 2005, pp. 641-644.

[47] G. Ascia, V. Catania, and M. Palesi, "A GA Bsed Design Space Exploraton Framework for Parameterized System-O-A-Chip Platforms," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 4, pp. 329-345, August 2004.

[48] I. Erfurth and W. Rossak, "CUTA4UML - Bridging the Gap Between Informal and Formal Requirements for Dynamic System Aspects," in *Software Engineering, 2010 ACM/IEEE 32rd International Conference Volume 2*, Cape Town, 2010, pp. 171-174.

[49] H. M. Jani, "Applying Case-Based Reasoning to Software Requirements Specifications Quality Analysis System," in *In Proc. International Conference on Software Engineering and Data Mining (SEDM2010)*, 2007, pp. 140-144.

[50] I. Erfurth and W.R Rossak, "A Look at Difficulties in Practical Software Development from the Developers Perspective - A Field Study and a First Solution with UPEX.," in *Internation Conference and Workshops on the Engineering of Computer- Based Systems*, Washington, DC, 2007, pp. 241-248.

[51] B. H. Cheng and J. M. Atlee, "Research Directions in Requirements Engineering," in *International Conference on Software Engineering. FOSE '07 IEEE CS*, Washington, DC, 2007, pp. 285-303.

[52] A. I. Antón, "Goal Based Requirements Analysis," in *Procedings of the International Conference on Requirements Elicitation, 1996 IEEE*, Colorado Springs, CO, 1996, pp. 136-144.

[53] A. Sutcliffe, "Scenario-Based Requirements Engineering," in *Requirement Engineering Conference, 2003. Proceedings 11th IEEE*, 2003, pp. 320-329.

[54] R. Gong, S. Luo, and J. He, "Use Case Based Innovative Design of E-Commerce Website.," in *10th International Conference on Computer Aided Industrial Design and Conceptual Design(AAID) 2009, Proceedings IEEE*, WenZhou, 2009, pp. 2021-2025.

[55] A. Szóke, "A Proposed Method for Release Planning from Use Case-Based Requirements Specification.," in *34th Euromicro Conference on Software and Advanced Applications. Proceedings IEEE.*, Parma, 2008, pp. 449-456.

[56] A. Ebnenasir, B. H.C Chen, and S. Konrad, "Use Case-Based Modelling and Analysis of Failsafe Fault-Tolerance," in *14th IEEE International Requirements Engineering Conference (RE'06).* , Minneapolis/ St. Paul, MN, 2006, pp. 343-344.

[57] H. M. Jani and T. Islam, "A Framework of Software Requirements Quality Analysis System Using Case-Based Reasonong and Neural Network," in *2012, 6th International Conference on Information Science and Service Science and Data Mining (ISSDM), Proceeding IEEE*, Taipei, 2012, pp. 152-157.

[58] M. B Bashir, S. A Latif, A. A Ahmed, A. Yousef, and M. E Eltayeeb, "Content-Based Information Retrieval Techniques Based on Grid Computing: A Review," *Institution of Electronics and Telecommunication Engineers, Technical Review* , vol. 30, no. 3, pp. 223-231, May-June 2013.

[59] T. W. Liao, Z. Zhang, and C. R. Mount, "Similarity Measures for Retrieval in Case-Based Reasoning Systems. ," in *Applied Artificial Intelligence,* , 1998, pp. 267-288.

[60] B. C Chatterjee, N. Sarma, and P. P Sahu, "Review and Performance Analysis on Routing and Wavelength Assignment Approaches for Optical Networks," *Institution of Electronics and Telecommunication Engineers, Technical Review*, vol. 30, no. 1, pp. 12-23, Jan-Feb 2013.

[61] Y. Lee and W. Zhao, "An Ontology-Based Approach for Domain Requirements Elicitation and Analysis," in *First International Multi-Symposiums on Computer and Computational Sciences, 2006. IMSCCS '06*, Hanzhou, Zhejiang, 2006, pp. 364-371.

[62] X. Zhu and Z. Jin, "Inconscistency Measurement of Software Requirements Specifications: An Ontology-Based Approach," in *Engineering of Complex Computer Systems, 2005. ICECCS 2005. Proceedings. 10th IEEE International Conference on*, 2005, pp. 402-410.

[63] F. Anwar and R. Razali, "A Practical Guide to Requirements Elicitation Techniques Selection - An Empirical Study," *Middle-East Journal of Scientific Research*, vol. 2, no. 8, pp. 1059-1067, 2012.

[64] Z. Zhang, "Effective Requirements Development - A comparison of Requirements Elicitation Techniques," in *Proceedings of Software Quality Management Conference, British Computer Society*, 2007, pp. 225-240.

[65] O. Dieste and N. Juristo, "Systematic review and aggregation of empirical studies on elicitation techniques," *Software Engineering, IEEE Transactions* , vol. 37, no. 2, pp. 283-304, March 2011.

[66] A. M. Hickey and A. M. Davis, "Requirements Elicitation and Elicitation Techniques Selection: A Model for Two Knowledge-Intensive Software Development Process," in *Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS' 03)*, Hawaii, 2003.

[67] H. N. Al-Duwaish, "Parameterization and Compensation of Friction Forces Using Genetic Algorithms," in *Industry Applications Conference, 1999. Thirty-Fourth IAS Annual Meeting. Conference Record of the 1999 IEEE*, Phoenix, AZ , 1999, pp. 653-655.

[68] N. G Jayalakshmi and H. S Vidya, "Educationally Effective Teaching of Design and Analysis of Algorithms.," in *IEEE international Conference in Innovation and Technology in Education (MITE)*, Jaipur, 2013, pp. 57-62.

[69] E. Chikohora and O. O. Ekabua, "A Genetic Approach to Parameterization of Feature Extraction Algorithms in Remote Sensing Images," *International Journal of Soft Computing And Engineering.*, vol. 4, no. 2, pp. 144-149, May 2014.

[70] P. B. Brazdil and C. Soares, "A Comparison of Ranking Methods for Classification Algorithm Selection," in *Machine Learning: ECML 2000*, R. L. Mántaras, Ed. Porto, Portugal: Springer Berlin Heidelberg, 2000, pp. 63-75.

[71] W. M. Spears and V. Anand, "A Study Of Crossover Operators In Genetic Operators," in *Methodologies for Intelligent Systems*, W. Z. Ras and M. Zemankova, Eds. Charlotte, N. C, USA: Springer Berlin Heidelberg, 1991, pp. 409-418.

[72] M. E. Famer, S. Bapna, and A. K. Jain, "Large Scale Feature Selection Using Modified Random Mutation Hill Climbing," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on (Volume:2)* , 2004, pp. 287-290.

[73] D. Jeyabharathi and A. Suruliandi, "Perfomance Analysis of Feature Extraction and Classification Techniques in CBIR," in *International Conference on Circuits, Power and Computing Technologies*, 2013, pp. 1211-1214.

[74] D Ze-Feng, Y Zhou-Ping, and X You-Lun, "High Probability Impulse Noise-Removing Algorithm Based on Mathematical Morphology," *IEEE Signal Processing Letters*, vol. 14, no. 1, pp. 31-34, January 2007.

[75] N. Y Khan, B. McCane, and G. Wyvill, "SIFT and SURF Performance Evaluation Against Various Image Deformations on Benchmark Dataset. ," in *International Conference on Digital Image Computing Techniques and Applications*, Noosa, QLD, 2011, pp. 501-506.

[76] N. M Suaib, M. H Marhaban, M. I Saripan, and S. Anom Ahmad, "Performance Evaluation of Feature Detection and Feature Matching for Stereo Visual Odometry using SIFT and SURF ," in *2014 IEEE Region 10 Symposium*, Kuala Lumpur, 2014, pp. 200-203.

[77] S. O Kuznetsov and S. A Obiedkov , "Comparing performance of algorithms for generating concept lattices," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 14, no. 2-3, pp. 189-216, 2002.

[78] M. B Bashir, M. A Latiff, A. A Ahmed, A. Yousif, and M. E Eltayeeb, "Content-based Information Retrieval Techniques Based on Grid Computing: A review," *The Institute of Electronics and Telecomunication Engineers*, vol. 30, no. 3, pp. 223-232, May 2013.

[79] S. G. Devi, K. Selvan, and S. P. Rajagopalan, "An Abstract to Calculate Big O Factors of Time and Space Complexity of Machine Code," in *International Conference on Sustainable Energy and Intelligent Systems (SEISCON 2011).*, Chenai, 2011, pp. 844-847.

[80] A. Norouzi , M. M Rahim, A. Altameem , T. Saba, and A. E Rad , "Medical Image

Segmentation Methods, Algorithms and Applications," *The Institute of Electronics and Telecommunication Engineers* , vol. 31, no. 3, pp. 199-213, May 2014.

[81] M. B. Fayek, "The suitable Fitness Test-bed for Computing Candidates in GA," in *Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligence Agents*, Vienna, 2005, pp. 100-106.

[82] Q. Lindi, "A Review Of Techniques For Extracting Linear Features From Imagery," *Photogrammetric Engineering & Remote Sensing*, vol. 70, pp. 1383–1392, December 2004.

[83] L. Najman and H. Tablot, "Introduction To Mathematical Morphology," in *Mathematical Morphology*, L. Najman and H. Tablot, Eds. London, United Kingdom: ISTE Ltd and J. Wiley and Sons, 2010, ch. 1, pp. 3-34. [Online]. http://books.google.co.za/books?id=_i8mRajKMfMC&pg=PA155&lpg=PA155&dq=Chapter+6+Mathematical+morphology&source=bl&ots=2ipJT82EJz&sig=tv39djJby_CEdAL3b96u-sRtgy0&hl=en&sa=X&ei=H_xkUu_oI8rJhAf4oYH4Dg&ved=0CEgQ6AEwBQ#v=onepage&q=Chapter%206%20Mathematical%20, [accessed Aug. 2015]

[84] D. Hommes, S. J. Ross, F. J. Vesecky, and J. Daida, "Extracting Curvilinear Features From Synthetic Aperture Radar Images Of Arctic Ice: Algorithm Discovery Using The Genetic Programming Paradigm," *International GeoScience and Remote Sensing Symposym*, pp. 10-14, July 1995.

[85] D. Howard, "Top Down Modelling With Genetic Programming," *Biocomputing Developmental Systems Group*, vol. LNAI 3215, pp. 217-223, 2004.

[86] F. Bianconi and A. Fernandez, "Evaluation of the effects of Gabor Filter Parameters on Texture Classification.," *Pattern Recognition*, vol. 40, no. 12, pp. 3325-3335, 2007.

[87] S. S Morade and S. Patnaik, "A Genetic Algorithm-Based 3D Feature Selection for Lip Reading.," in *International Conference on Pervasive Computing (ICPC 2015)*, 2015.

[88] E. Chikohora and O. O Ekabua, "Feature Extraction Techniques in Remote Sensing Images: A survey on Algorithms, Parameterization and Perfomance," *Internation Journal of Soft Computing and Engineering*, vol. 4, no. 1, pp. 140-144, March 2014.

[89] C. M. Keet. (2002, May) Homepage of Maria (Marijke) Keet : Genetic Algorithms - An Overview. [Online]. Http://www.meteck.org/gaover.html [accessed 14 June 2014]

[90] S. Cateni, M. Vannucci, M. Vannocci, and V. Colla, "Variable Selection and Feature Extraction Through Artificial Intelligence Techniques," in *Multivariate Analysis in Management, Engineering and the Sciences*, L. Valim de Freitas and A. P. Barbosa Rod,

Eds. Ghezzano, Pisa, Italy: INTECH, 2013, ch. 6.

[91] V. Khorani, N. Forouzideh, and A. M Nasrabadi, "Artificial Neural Network Weights OptimizationUsing ICA, GA, ICA-GA and R-ICA-GA: Comparing Perfomances," in *Hybrid Intelligent Models and Applications (HIMA 2011 IEEE)*, Paris, 2011, pp. 61-67.

[92] L. Wei, Y. Pu-liu, X. De-ling ; Zhou Cong , and Z. Cong , "An approach to dynamic fingerprint image enhancement," in *Multi-Agent Security and Survivability, 2004 IEEE First Symposium*, 2004, pp. 294-297.

[93] O. Mendoza, Melin P., and O. Castillo, "Neural Networks Recognition Rate as Index to Compare the Performance of Fuzzy Edge Detectors," in *Neural Networks (IJCNN), the 2010 International Joint Conference* , Barcelona, 2010, pp. 1-6.

[94] K. C. Chung, S. C. Kee, and S. R. Kim, "Face Recognition Using Principal Component Analysis of Gabor Filter Responses," in *Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, 1999. Proceedings. International Workshop on*, Corfu, 1999, pp. 53-57.

[95] G. J Edwards, T. F Cootes, and C. J Taylor, "Face Recognition using Active Appearance Models.," in *Proceedings of the European Conference in Computer Vision.*, 1998, pp. 581-695.

Appendices

Appendices A and B gives some illustrations and explanations on the simulations that we did as part of our experiments to test the performance of the GenApp and justify the effect of parameterization when using feature extraction techniques such as the Gabor filter.

Appendix A: GenApp Simulations and Results

Figure A1 shows the genAID simulation tool's input interface with the values that we used to simulate the GenApp.

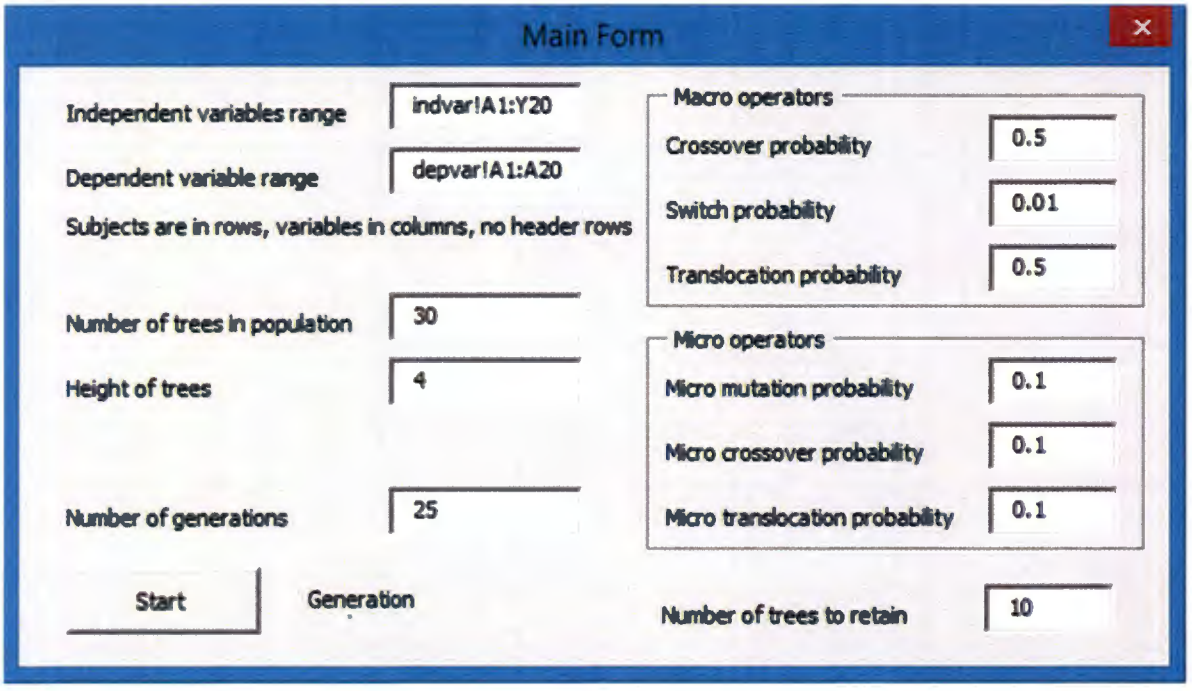


Figure A1: Simulation Input Form

Figure A2 shows the major parameters used to calculate the fitness values of the GenApp and the decision trees that were constructed while optimizing parameter values of the GenApp.

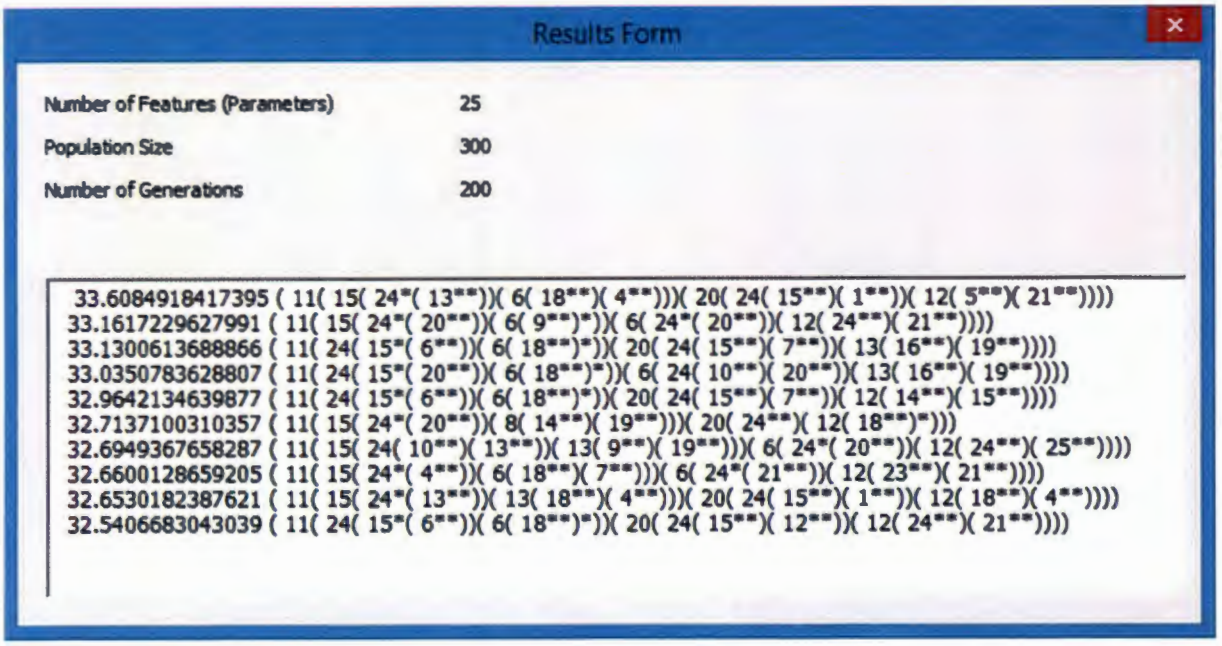


Figure A2: Simulation Results Form

Figure A3 shows the data in phenotype that was used for simulation. The data was obtained from the random number generator (RNG) which was considered as our dependent variables.

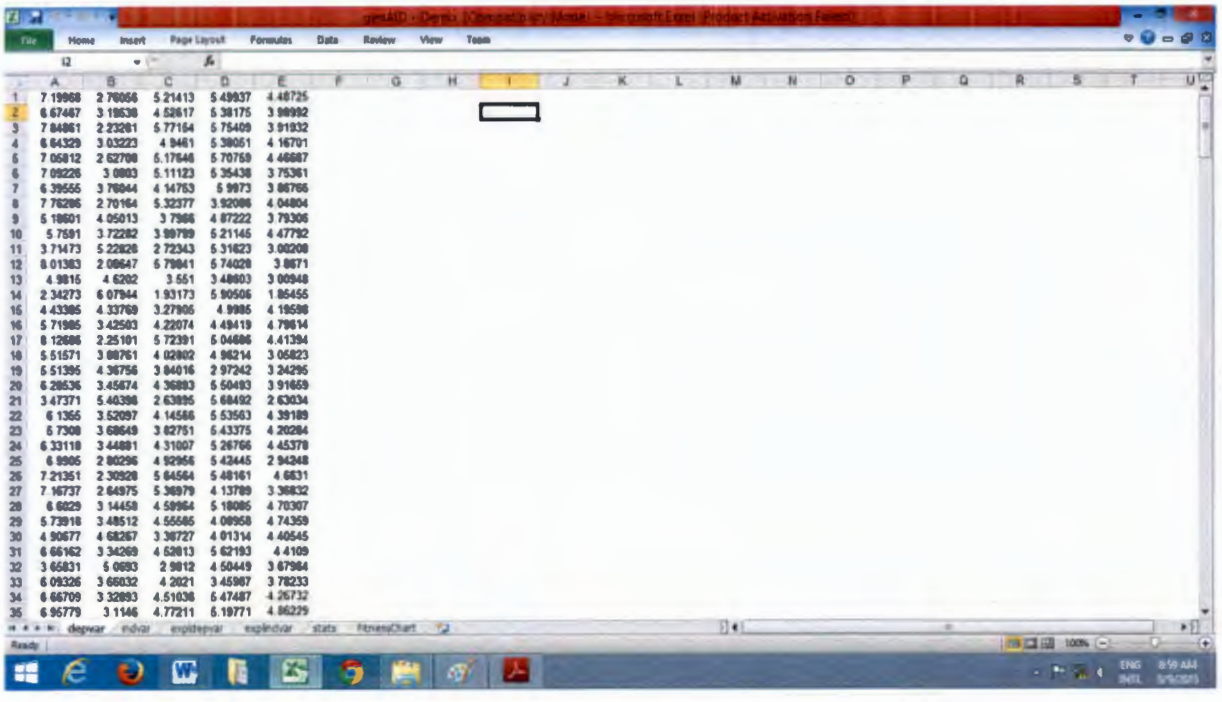


Figure A3: Dependent Variables in Phenotype

Figure A4 shows the data in genotype that was used for simulation. The data was obtained through the dependent variables dependent values using the two's complement method.

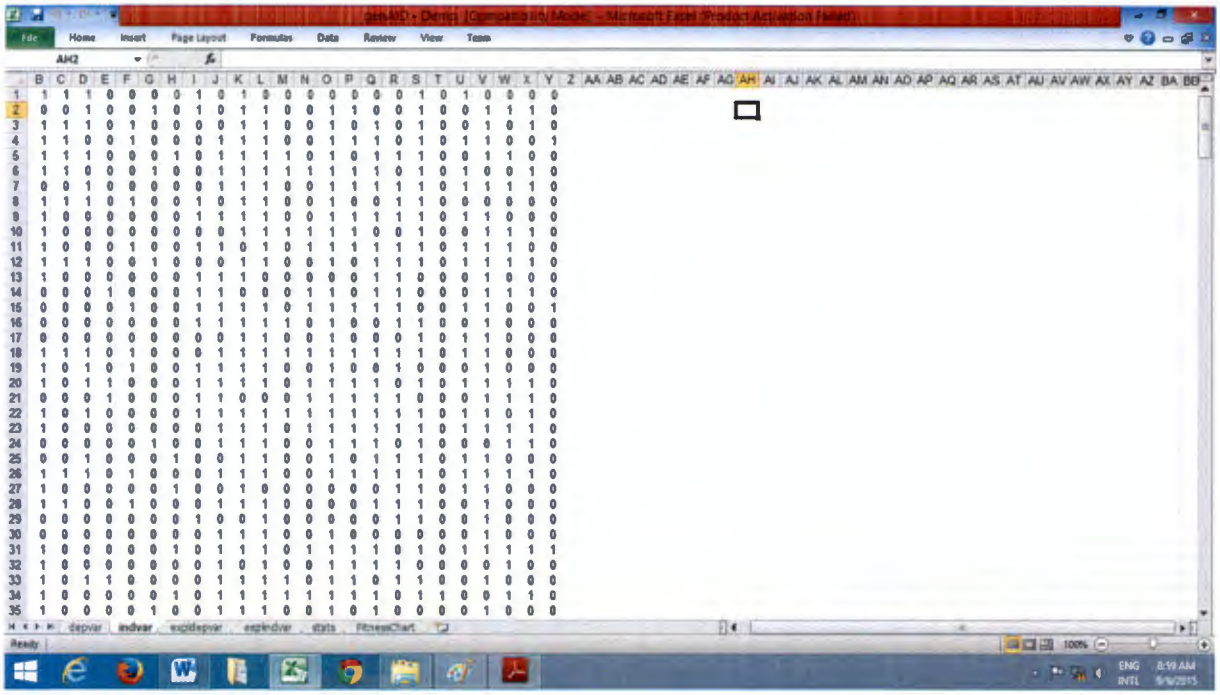
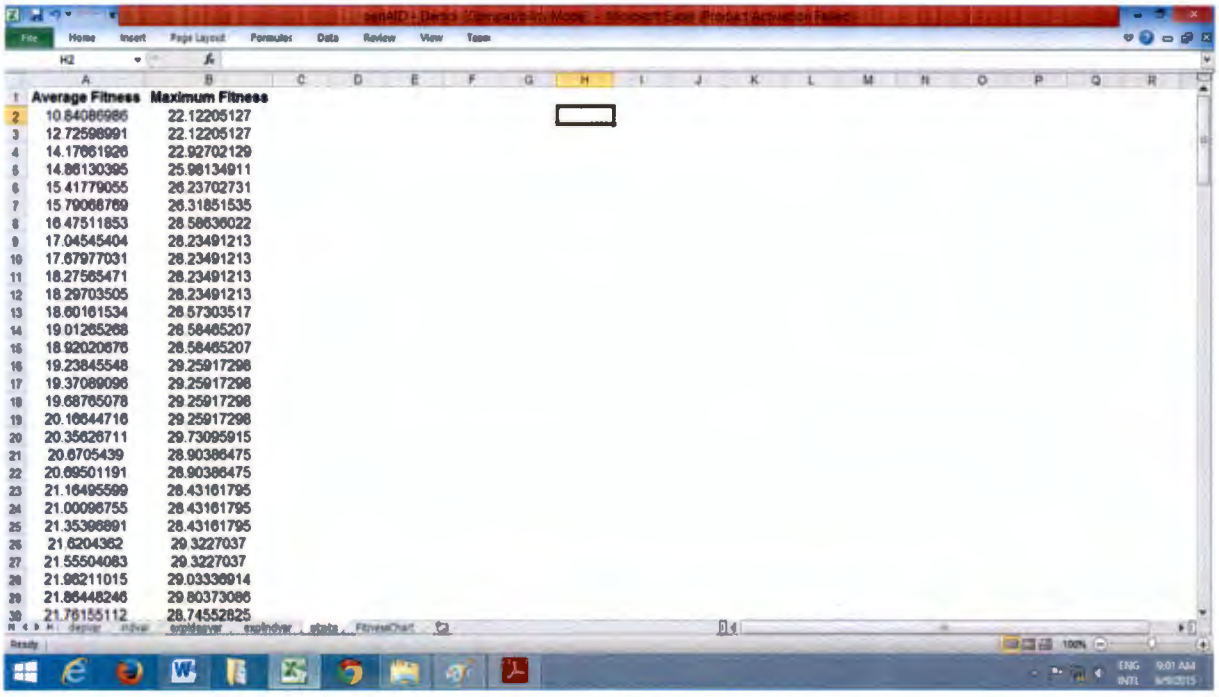


Figure A4: Dependent Variables in Phenotype

Figure A5 and A6 shows the fitness values obtained from the simulator and their presentation graphically. We had to consider the maximum and average fitness values from the simulator after discovering some tendencies of the simulator to converge to local minima resulting in prolonging the process of optimizing the parameter values.



Average Fitness	Maximum Fitness
10.84086986	22.12205127
12.72598991	22.12205127
14.17661926	22.92702129
14.86130395	25.98134911
15.41779055	26.23702731
15.79068769	26.31851535
16.47511853	28.58636022
17.04545404	28.23491213
17.67977031	28.23491213
18.27565471	28.23491213
18.29703505	28.23491213
18.60161534	28.57303517
19.01265268	28.58465207
18.92020676	28.58465207
19.23845548	29.25917298
19.37089096	29.25917298
19.68765078	29.25917298
20.16644716	29.25917298
20.35626711	29.73095915
20.6705439	28.90386475
20.69501191	28.90386475
21.16495599	28.43161795
21.00096755	28.43161795
21.35396891	28.43161795
21.6204362	29.3227037
21.55504083	29.3227037
21.98211015	29.03336914
21.86448246	29.80373086
21.76155112	28.74552825

Figure A5: Fitness Values

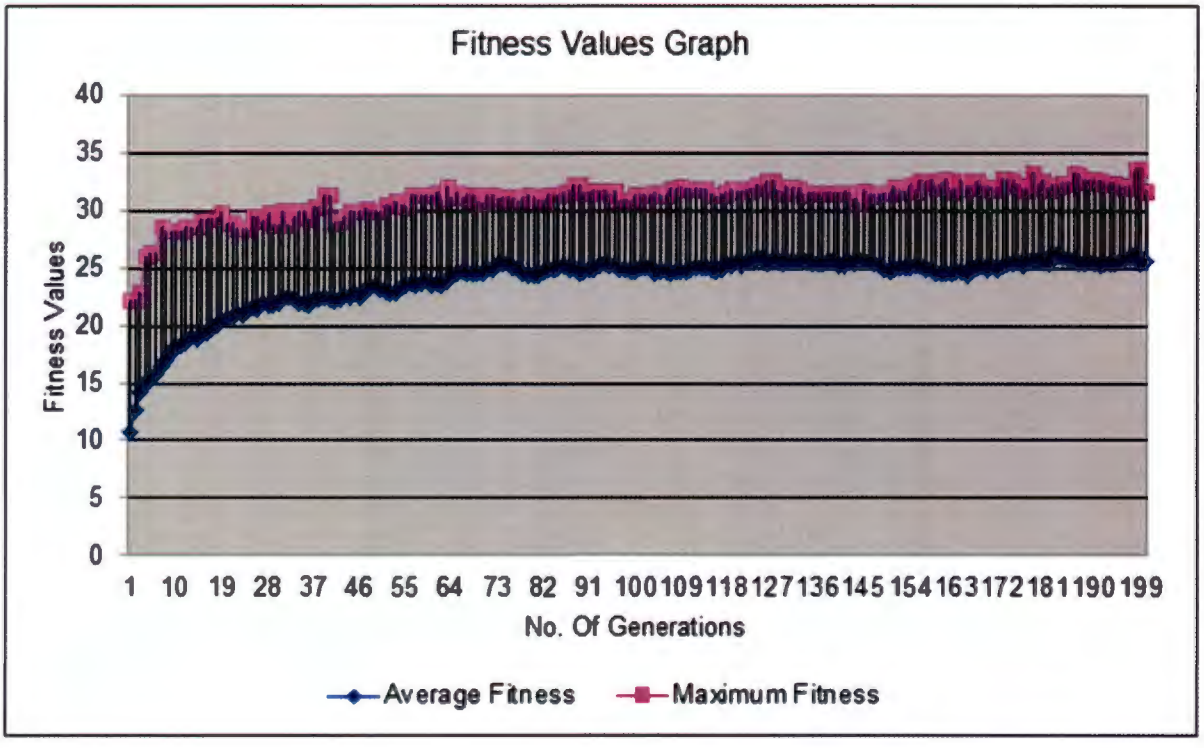


Figure A6: Fitness Values Graph

Appendix B: Gabor Filter for Image Processing

Figure A7 shows the simulation interface that we used to simulate Gabor filter on the effects of the input parameter values on the resulting image. Wavelength, orientation, aspect ratio, bandwidth and the number of orientations are the parameters being monitored in this instance.

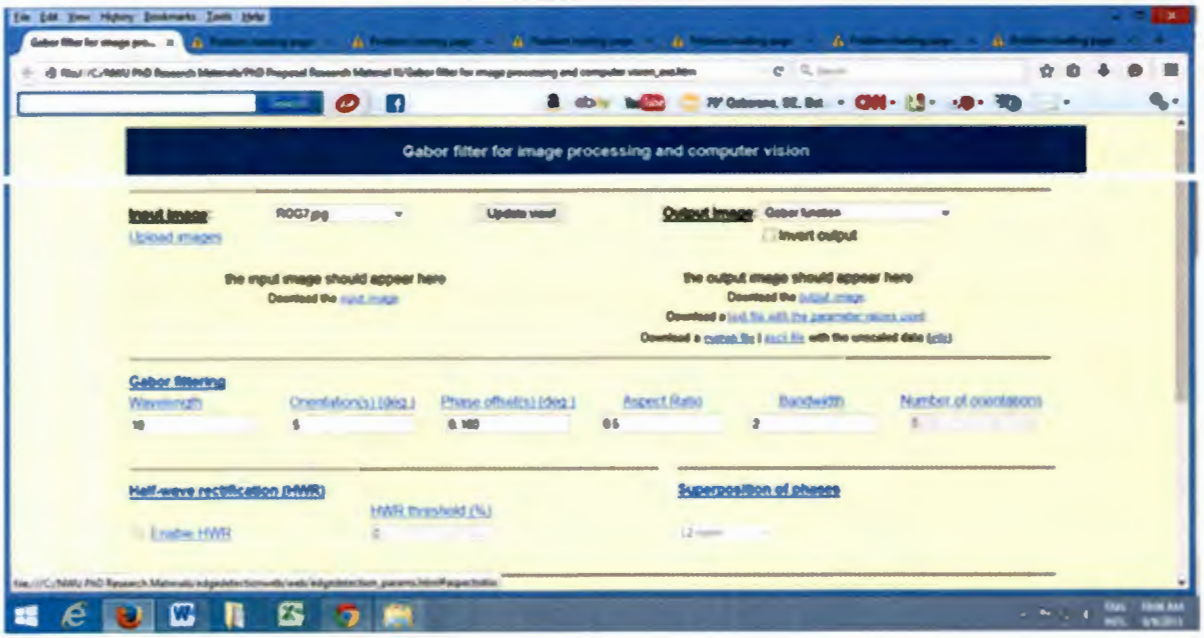


Figure A7: The Simulation Interface for the Gabor Filter