STATUS OF THESIS

Title of thesis	VISUALIZATION OF GENETIC ALGORITHM ON 2-D GRAPH TO ACCELERATE THE SEARCHING WITH HUMAN INTERVENTIONS

I HUMERA FAROOQ

hereby allow my thesis to be placed at the Information Resource Center (IRC) of Universiti Teknologi PETRONAS (UTP) with the following conditions:

1. The thesis becomes the property of UTP

2. The IRC of UTP may make copies of the thesis for academic purposes only.

3. This thesis is classified as

[ ] Confidential

[✓] Non-confidential

If this thesis is confidential, please state the reason:

____

____

____

The contents of the thesis will remain confidential for __________ years.

Remarks on disclosure:

____

____

____

Endorsed by

[illegible]

Signature of Author

[illegible]

Computer & Information Scien
Universiti Teknologi PETRONA
31750 Tronoh
Perak Darul Ridzuan, MALAYS

Signature of Supervisor

Permanent address: H.No.317, St No. 11, Mazhar Farid Colony, Sadiq Abad. Pakistan.

Name of Supervisor
Dr. Mohammad Nordin Zakaria

Date : 02/02/2012

Date: 2/2/12

UNIVERSITI TEKNOLOGI PETRONAS

VISUALIZATION OF GENETIC ALGORITHM BASED ON 2-D GRAPH TO ACCELERATE THE SEARCHING WITH HUMAN INTERVENTIONS.

by

HUMERA FAROOQ

The undersigned certify that they have read, and recommend to the Postgraduate Studies Programme for acceptance this thesis for the fulfilment of the requirements for the degree stated.

Signature:	[illegible] Computer & Information Sciences Universiti Teknologi PETRONAS 31750 Tronoh Perak Darul Ridzuan, MALAYSIA
Main Supervisor:	Dr. Mohammad Nordin Zakaria
Signature:	[illegible] Assoc. Prof. Dr. Mohd Fadzil Bin Hassan Head Department of Computer & Information Sciences Universiti Teknologi PETRONAS
Co-Supervisor:	Assoc. Prof. Dr. Mohd Fadzil Hassan
Signature:	[illegible] Dr Suziah Sulaiman Senior Lecturer Computer & Information Sciences Department Universiti Teknologi PETRONAS
Co-Supervisor:	Dr. Suziah Sulaiman
Signature:	[illegible] Assoc. Prof. Dr. Mohd Fadzil Bin Hassan Head Department of Computer & Information Sciences Universiti Teknologi PETRONAS
Head of Department:	Assoc. Prof. Dr. Mohd Fadzil Hassan
Date:	3/2/2012

VISUALIZATION OF GENETIC ALGORITHM BASED ON 2-D GRAPH TO ACCELERATE THE SEARCHING WITH HUMAN INTERVENTIONS.

by

HUMERA FAROOQ

A Thesis

Submitted to the Postgraduate Studies Programme

as a Requirement for the Degree of

DOCTOR OF PHILOSOPHY

COMPUTER AND INFORMATION SCIENCES

UNIVERSITI TEKNOLOGI PETRONAS

BANDAR SERI ISKANDAR,

PERAK

FEBRUARY 2012

DECLARATION OF THESIS

Title of thesis: VISUALIZATION OF GENETIC ALGORITHM BASED ON 2-D GRAPH TO ACCELERATE THE SEARCHING WITH HUMAN INTERVENTIONS.

I HUMERA FAROOQ

hereby declare that the thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at UTP or other institutions.

Witnessed by

[illegible]

Signature of Author

Permanent address: H.No.317,St No.11, Mazhar Farid Colony, Sadiq Abad. Pakistan.

Date : 02/02/2012

[illegible]

Computer & Information Scienc
Universiti Teknologi PETRONAS
31750 Tronoh
Perak Darul Ridzuan, MALAYSI

Signature of Supervisor

Name of Supervisor
Dr. Mohammad Nordin Zakaria

Date: 2/2/12

TO MY DAUGHTERS, ATIKAH AND FATIMAH TARIQ

ACKNOWLEDGEMENT

I realize I wouldn't have reached my destination without the help of several people. First and foremost, I am deeply indebted to my supervisor Dr Mohammad Nordin Zakaria. Throughout my research, he provided invaluable guidance, skilled advice and incredible patience. His guidance provided me with much confidence to accomplish my research successfully. I would also like to appreciate my co-supervisors Dr Suziah Sulaiman and Dr Mohd Fadzil Hassan for their assistance and guidance for writing and fruitful discussions.

I would like to thank Universiti Teknologi PETRONAS for granting me a Graduate Assistantship. Also, thanks to the administrative staff of the Department of Computer and Information Sciences and the Post Graduate Department who rendered their every possible help whenever needed.

I am extremely grateful to my family who have prepared me for this long-lasting trek. The successful completion of my PhD studies is the fruit of their sacrifices, their devotion and their determination. My parents, sisters and my husband deserve much of the credit. Special thanks to Dr. Rahat Iqbal for his constant moral support in finalizing my research. My endeavours would not have been successful without them.

ABSTRACT

The Genetic Algorithm is an area in the field of Artificial Intelligence that is founded on the principles of biological evolution. Visualization techniques help in understanding the searching behaviour of Genetic Algorithm. It also makes possible the user interactions during the searching process. It is noted that active user intervention increases the acceleration of Genetic Algorithm towards an optimal solution.

In proposed research work, the user is aided by a visualization based on the representation of multidimensional Genetic Algorithm data on 2-D space. The aim of the proposed approach is to study the benefit of using visualization techniques to explorer Genetic Algorithm data based on gene values. The user participates in the search by proposing a new individual. This is different from existing Interactive Genetic Algorithm in which selection and evaluation of solutions is done by the users. A tool termed as VIGA-2D (Visualization of Genetic Algorithm using 2-D Graph) is implemented to accomplish this goal. This visual tool enables the display of the evolution of gene values from generation to generation to observing and analysing the behaviour of the search space with user interactions. Individuals for the next generation are selected by using the objective function. Hence, a novel human-machine interaction is developed in the proposed approach.

The efficiency of the proposed approach is evaluated by two benchmark functions. The analysis and comparison of VIGA-2D is based on convergence test against the results obtained from the Simple Genetic Algorithm. This comparison is based on the same parameters except for the interactions of the user. The application of proposed approach is the modelling the branching structures by deriving a rule from best solution of VIGA-2D. The comparison of results is based on the different user's perceptions, their involvement in the VIGA-2D and the difference of the fitness convergence as compared to Simple Genetic Algorithm.

ABSTRAK

Algoritma Genetik ialah satu bidang dalam Kecerdasan Buatan yang diasaskan pada prinsip-prinsip evolusi kajihayat. Teknik-teknik visualisasi dapat membantu dalam memahami perilaku carian algoritma genetik. Ia juga membolehkan interaksi pengguna ketika proses pencarian. Perlu dinyatakan bahawa intervensi pengguna secara aktif dapat meningkatkan pencepatan algoritma genetik ke arah penyelesaian optima.

Di dalam kajian yang dicadangkan, pengguna dibantu oleh visualisasi berdasarkan perwakilan data multi-dimensi pada ruang 2-D. Tujuan pendekatan yang dicadangkan adalah untuk mengkaji manafaat yang diperolehi apabila menggunakan teknik-teknik visualisasi untuk meneroka data algoritma genetik berdasarkan nilai-nilai genetik. Pengguna menyertai pencarian tersebut dengan mencadangkan individu baru. Ini berbeza dari algoritma genetik interaktif yang sedia ada dimana pilihan dan penilaian dilakukan sendiri oleh pengguna. Alatan yang digelar sebagai VIGA-2D (Visualisasi Algoritma Genetik menggunakan 2-D Grafik) telah dilaksanakan untuk mencapai matlamat ini. Alatan visualisasi ini membolehkan paparan evolusi nilai gen dari generasi ke generasi untuk mengamati dan menganalisasi perilaku ruangan carian dengan interaksi pengguna. Individu untuk generasi seterusnya dipilih dengan menggunakan fungsi objektif. Oleh kerana itu, terciptalah interaksi manusia-mesin baru dalam pendekatan yang.

Kecekapan dari pendekatan yang dicadangkan dinilai dengan ujian konvergensi dan analisis berdasarkan purata dan kecergasan terbaik dari setiap generasi. Kemudian, hasil yang diperolehi daripada VIGA-2D dibandingkan dengan Simple Genetic Algorithm (SGA) di bawah parameter yang sama kecuali untuk interaksi pengguna. Perbandingan keputusan didasarkan pada persepsi pengguna yang berbeza, penglibatan mereka dalam-VIGA 2D, dan perbezaan dari konvergensi kecergasan.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

GA	Genetic Algorithm
SGA	Simple Genetic Algorithm
IGA	Interactive Genetic Algorithm
AI	Artificial Intelligence
2-D	2 Dimensional
3-D	3 Dimensional
n-D	Multi Dimensional
VIGA-2D	Visualization of Genetic Algorithm on 2-D Graph
TIGA	Traditional Interactive Genetic Algorithm
RGB	Red, Green , Blue
L-System	Lindenmayer System

LIST OF SYMBOLS

k	Generation
m	Size of population
l	Length of chromosome
v_{unit}	Pixel value range along vertical view
V_{length}	Vertical length of graph
m_{max}	Maximum range of genes value
G_{value}	Genes value to display
m_{min}	Minimum range of genes value
N_{child}	Proposed individual
$F_{average}$	Average fitness of current generation
P_{size}	Size of population before population
P_{size}'	Size of population after interaction
E_{pop}	Existing population
E_{pop}'	Size of new population
y_{pixel}	y-axis position at graph
h_{unit}	Pixel value along horizontal view
h_{length}	Horizontal length of graph,
m_{value}	Minimum range of fitness
$T_{fitness}$	Total fitness
p_m	Selection probability
q_m	Cumulative probability
$S_{difference}$	Square difference of two arrays
$Gparam$	Genetic Algorithm generated solution
$Oparam$	Target solution
f	Fitness value of the current evolving individual
n	Length of input chromosome
m_t	Mutation rate

CHAPTER 1

INTRODUCTION

1.1 Background

Computer graphics always aided the researchers in exploring complicated problems visually. Visualization is a technique used to represent abstract data in visual form. This technique helps the user to explore, interact, reorganize and understand the meaning of complicated or hidden data. Visualization plays an important role in helping the users to understand the problems graphically [1]. It raises the level of understanding for multidimensional data. Another advantage of using this technology is a reduction of the work period. Visualization is commonly known as a method of seeing the unseen [2]. Visualization does not mean only viewing the graphical pictures, but it also includes analysing and interpreting the data [3].

The visualization technique also facilitates the development of systems and evaluation of data in different applications [4]. This technique also helps to visualize the internal process of algorithms which uses black box strategy (only input and output parameters are known), to understand their internal functionality and behaviour.

Over the last few years, scientists have begun taking a keen interest in applying the visualization techniques for the search space of Genetic Algorithm (GA). These techniques are applied to adapt the dynamic changes in the search process and to introduce user involvement. In other words, GA may develop as an interactive tool with different visualization techniques. This technique is known as the Interactive Genetic Algorithm (IGA). The aim of such visualization is to examine the behaviour of GA during the evolution process, help with convergence towards an optimal solution, or to explore the search process of GA to get more than one best solutions.

Moreover, the interaction with a user helps this algorithm to converge efficiently and in fewer generations towards an optimal solution.

The GA was proposed by John Holland in 1960's , based on the adaptive process of natural systems [5]. GA is a dynamic random searching algorithm, which gained massive popularity in very little time because of its effectiveness in solving difficult optimization problems. It is described as a search technique, applied in the computation evolution to find exact or approximate solutions to optimization and searching problems. It is also known as the global optimization algorithm; moreover, it also works well on noisy functions having many local optima. The searching mechanism of this algorithm is often known as the blind search method because it does not require any information about the first derivative or any other restrictive assumption before solving a problem. Unlike other techniques of Artificial Intelligence (AI), GA is more robust (error free), even in the presence of small noise or any small change in the input; it does not break easily. The direct search method and robustness quality makes this algorithm more favourable as compared to other searching techniques [5, 6].

Techniques used by GA are based on natural methods and biological genetics to produce the solutions(next population), and it follows the Darwin theory of natural selection methods [7, 8]. Therefore, it uses biological terminology to express things. The problem is introduced to GA in the genotypic form termed as chromosome encoding. Chromosome is a kind of data structure consisting of genes. Each gene encodes a parameter or value of the problem, which is being evolved by the GA; these are termed as gene values. Construction of this chromosome depends on the specified problem. GA works on the population of the individual; in other words, a group of individuals is known as a population. It is considered as a tool for solving optimization and searching problems in which the result is a population of solutions not an individual solution [9]. These solutions (individuals) compete for survival in the population in each generation. The better the solution, the higher the chance it has to survive for next generation.

The GA generates a population by selecting random individuals according to their fitness. Fitness is a value which determines the optimal individual in the current generation to be selected as a parent for the next generation. In other words, it is a value which determines the performance of GA towards best solution. The fitness function varies from problem to problem. Individuals go through a process of selection after assigning the fitness value for the survival of the fittest. This cycle continues until the stopping criterion is met.

1.2 Thesis Motivation

Visualization techniques make it possible to create a visual environment for understanding the biologically inspired methods in a better way [11]. From last few years scientist are taking keen interest to use visualization techniques to explorer GA search space [12, 13]. This visualization environment makes it possible to view, interact and understand the searching process of GA [15]. In this way, different visualization techniques make it possible to analyse GA solutions in search space and to bring some changes in them which was quite difficult in Simple Genetic Algorithm (SGA). Also these visualization techniques help to understand the GA convergence and to change the parameters or functions during this search process [10]. Moreover, artificial selection and evaluation of solutions by the user, help GA to search the desired solution in fewer numbers of generations [14]. Thus, the human interaction can accelerate the searching process of GA for complex problems easily [15].

IGA is an extension of GA in which human interaction and evaluation is needed to get a solution. This interactive approach of GA is widely used for creative applications such as architecture, art, music and design. In IGA, different computer graphic techniques are used for solving complex problems. In most of these applications, solutions are represented as 2 or 3-D models [16]. Hence, visualization makes it easy to choose different parameters and functions, and get a solution in fewer generations. Furthermore, with this technique, the exploration of search space at the user level brings more variety of solutions [17]. It is often used to give a facility to the user for exploring the search space for other solutions instead of only optimum solution. Drawback of these existing applications lies in their continuous dependence on the user for performing search and evaluation for the fitter solution. Hence this continues search often create a tiresome environment for users. Another limitation of these applications is small number of generations. Furthermore, in these existing applications the individuals of search space are represented as 2 or 3 model. In this way, the existing visualization techniques are unable to determine and observe the searching behaviour of GA during evolution process.

The work done in this thesis is to aid the scientists and researchers in understanding the searching behaviour of GA by projecting the multidimensional GA data on lower dimension. This visualization is based on displaying gene values of each generation. Furthermore, for accelerating the searching process of GA, an idea of proposing a new individual into current generation has been introduced. Beside the existing applications, in proposed approach human interventions are not forced in every generation. The user may interact in any generation during the searching process to propose a new individual to be evolved in the next generation. In this way the proposed approach in this thesis is to understand and analyse the convergence of GA on 2-D graph and to do various interactions towards the best solution with the visualization technique as shown in Figure 1.1.

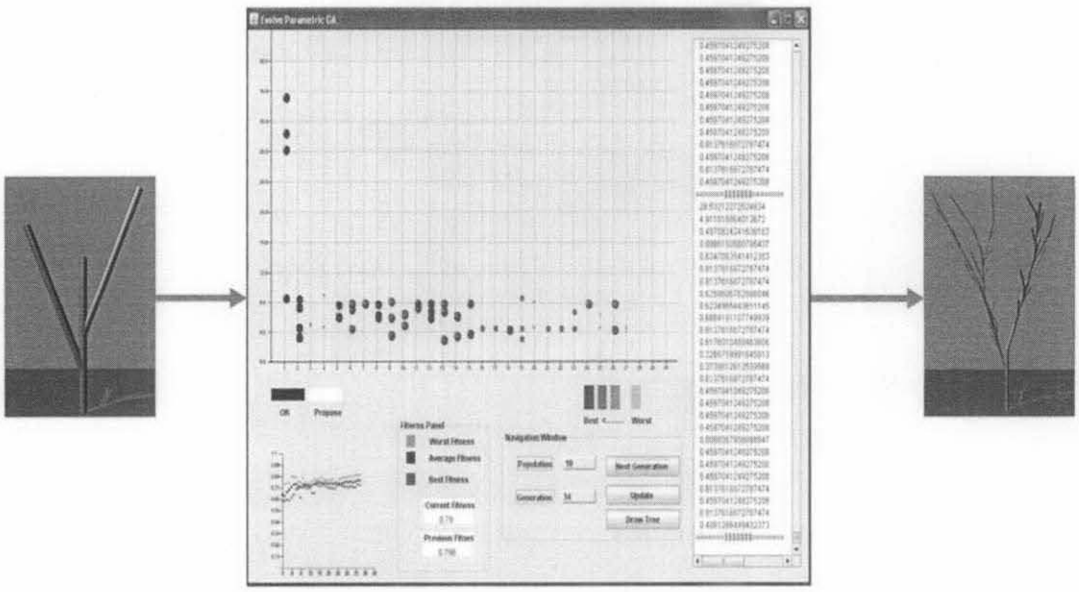


Figure 1.1: An Interactive Visualization for Modelling of Branching Structures

1.3 Problem Statement

Although Simple Genetic Algorithm (SGA) uses its operators and methods to search for an optimized solution in a multidimensional (n-D) search space more efficiently as compared to a human, however, humans have an excellent ability to examine and analyse the entire distribution of individuals in the visualized form that cannot be interpreted by the SGA [15]. That is why the IGA combines the algorithmic SGA search of the multidimensional (n-d) search space with the human global search in a mapped 2-D or 3-D space to represent the GA data [15, 21].

Initially, the techniques for visualisation of multidimensional data of GA were based on the combined search space or visualization of individuals on 3 or 2-D space [19]. Furthermore, these existing techniques for visualization of multidimensional data were only for information purpose [18]. In the visualization window, user may view the selected individuals [15] or combined search space on 3 or 2 or 1-Dimension but cannot interact with the search space [19]. In this way, with these existing applications, there is no user level exploration at the internal structure of chromosomes (gene values) [18, 20]. In addition, there was no user level understanding for the convergence and behaviour of GA used for searching.

On the other hand IGA is found to be a flexible and user friendly technique based on modelling the individuals of search space. It is used for solving artistically and atheistically relevant problems. Most of the Traditional Interactive Genetic Algorithm (TIGA) applications require human evaluation and selection in each generation. Based on the fitness assigned by the user, individuals are selected for the next generation. The main problem of TIGA is the continuous interaction of the user which causes fatigue to the user and gives unwanted solutions [15, 21, 22]. In addition, users can only select or deselect the solution or assign fitness to them but cannot change the parameters (gene values). Some main drawbacks of using TIGA are:

a) In existing applications with the TIGA, individuals of a generation are modelled and presented as 2 or 3D model. Only a few parameters may be changed by the user, i.e. colour, rotation or scaling [21].

b) In TIGA, fitness depends on the user i.e. the user evaluation and selection is used to create individuals for the next generation [21, 22]. This feature is only applicable to artistic and aesthetic problems, where the user's interest is more towards assigning fitness and evaluating solutions [22], but not on understanding the convergence behaviour of the GA.

c) These existing techniques with TIGA have fewer numbers of generations to prevent the user‘s fatigue. Moreover, a small search space is used for evolving individuals [22, 23].

The above limitations affect the performance of IGA, especially when it is applied to a problem with a large search space or larger number of generations [24]. For instance, the evaluation and selection in each generation may create user fatigue. Moreover, due to adopting a model view for the solutions, it is also unable to visualize the internal structure of the chromosome (gene values).

Since the user fatigue was a major problem in the TIGA, several techniques has been proposed to address this problem. The problem of assigning fitness was resolved by introducing a discrete fitness value [25] or an approximation technique [17] with the GA process. Although these proposed approaches successes to resolve the problem of the assigning fitness by user in each generation, however several other techniques were used to evaluate the fitness i.e. user has to spend a particular time on each individual [26] for assigning fitness or the user need to select some best individuals from each generation [15] or the fitness is restricted to some parameters. Moreover the selection of parent was also depends on the user. Furthermore these existing techniques work with the model based visualization that makes them unable to represent the distribution of gene values of the search space.

In the present work, we invited the user to directly participate in searching. Visualization of GA is carried out based on the gene values of the current generation and user interaction is only required after several generations. Besides the selection of the best individual or parents for the next generation, interference of the user is required to propose a new individual in any generation as shown in Figure 1.2. In this way, the proposed approach saves the user from the tiresome work of selection and evaluation of solutions in every generation. A human-machine interaction is used in which an objective function combines with IGA to assign fitness and GA selection method is used to select the parents for next generation. Hence, the active user interaction leads to a faster search, resulting in fitter solutions in fewer generations. The next section discusses research objectives of this thesis.

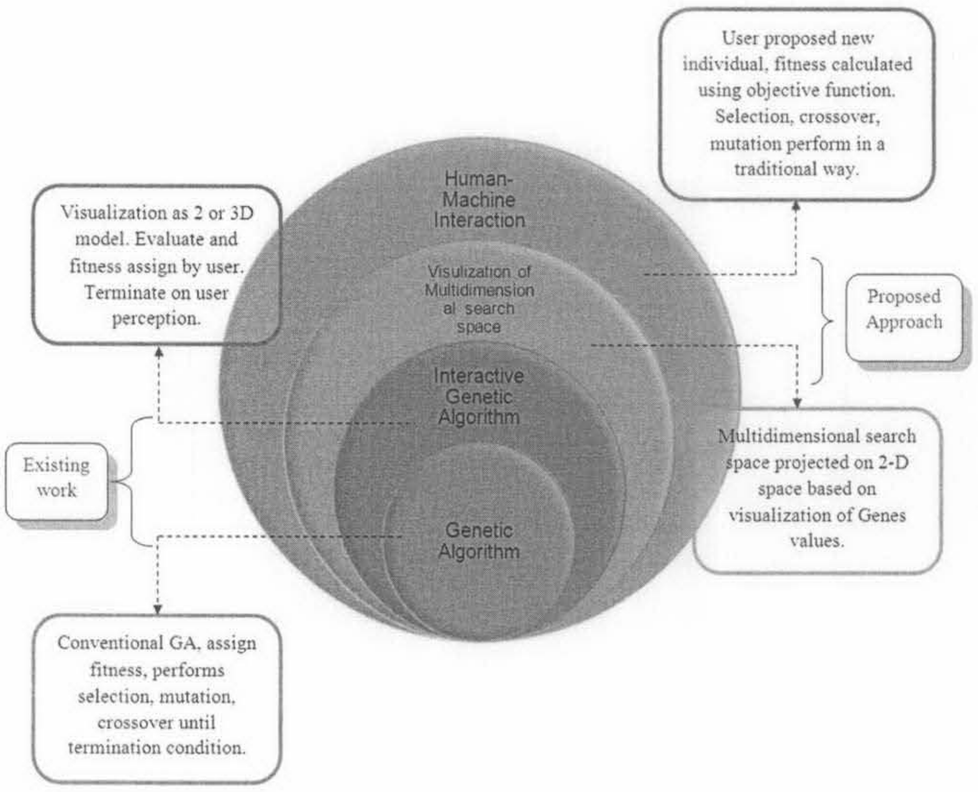

User proposed new individual, fitness calculated using objective function. Selection, crossover, mutation perform in a traditional way.
Visualization as 2 or 3D model. Evaluate and fitness assign by user. Terminate on user perception.
Human-Machine Interaction
Visulization of Multidimensional search space
Proposed Approach
Interactive Genetic Algorithm
Existing work
Multidimensional search space projected on 2-D space based on visualization of Genes values.
Genetic Algorithm
Conventional GA, assign fitness, performs selection, mutation, crossover until termination condition.

Figure 1.2: An Overview from Problem to Solution with the Proposed Approach

1.4 Research Objectives

The aim of this research is to investigate an approach that improves upon existing IGA techniques and to develop a technique for the user interaction during search process. The visualization of multidimensional GA data should be based on lower dimension to show the distribution of gene values onto the screen. This visualization will give a clear picture of the hidden process of GA involved for searching. Based on the aim of this research, the following research objectives are set:

1. To investigate a technique for visualization of multidimensional GA data onto the screen.
2. To propose a technique for user intervention with GA search space to accelerate its performance.
3. To develop an efficient and usable tool to achieve the above two objectives.
4. To investigate the application of the technique for modelling the growth process of branching structures.
5. To verify and validate the results of the proposed approach against the Simple Genetic Algorithm (SGA).

1.5 Thesis Contributions

The main contribution of this research is to propose an approach to visualize the multidimensional GA data and to improve the searching process of GA with a user's interventions in different generations. The advantage of the proposed technique is to reduce the user fatigue and to accelerate the searching ability of GA with the user interactions. This has been achieved by meeting all objectives set in the previous section. Additionally, an efficient and a usable visualization tool, VIGA-2D, has been developed for this research. This thesis has made the following contributions to the existing corpus of knowledge:

1. **Representation of Multidimensional Data Based on Gene Values:** To address the problem of representing the multidimensional data based on gene values, a 2-D graph visualization approach has been adopted. This visualization is based on representing each generation on a 2-D graph. In this 2-D graph, the Y-axis shows the gene values and the X-axis shows the genes' location for each generation.
2. **Accelerate the GA Performance with User Interventions:** To address the problem of accelerating the performance of GA with user interventions and

preventing the user fatigue as discussed in section 1.3, the interaction of the user is not forced in every generation. The user may go to several generations without interactions. For accelerating the performance of the proposed approach, the user interaction is involved to propose a fitter individual in current generation.

This individual becomes a part of the search space in the next generation. An objective function is used to calculate fitness of each individual. A dynamic population size is used to adjust the new individual into the search space, i.e. whenever a user proposes a new individual the size of the population is increased by one.

3. **Developing an Efficient and Usable Tool:** To accomplish the first two objectives, a tool is developed named VIGA-2D. This tool is developed with all important Graphical User Interface (GUI) components to show the performance of the proposed approach with different inputs. In this tool, the user may select different GA operators and rates for crossover and mutation. Two selection methods are implemented and it's the user choice to select any method. The visualization of fitness / generation 2-D graph is also a part of VIGA-2D. This graph helps to understand the convergence of GA towards an optimized solution. In order to generate branching structures from the output of VIGA-2D, a visualization window is implemented with all the important components. These components include rotation of the generated structure in 3-D space, shade, and the light effects and setting for background colour.

4. **Modelling the Growth Process of the Branching Structures:** The application of the proposed approach is the modelling of the growth process of the branching structures using VIGA-2D. To accomplish this goal, the Parametric L-System is used. For generating the branching structure L-System symbols are input by the user and the parameters of the L-System are evolved using VIGA-2D. It works in three stages. The first stage works with the input symbols, the second stage work with evolving parameters for Parametric L-System using VIGA-2D and in the third stage, the L-System rule generated with the output of VIGA-2D for modelling the further growth

process of the branching structures. The terminating condition for this application depends on the user's satisfaction.

1.6 Thesis Scope

The research effort made to understand the research presented in this thesis focuses on exploring the technique to visualize the multidimensional data of the GA search space. Based on the proposed method, a tool (VIGA-2D) is developed in which each generation is displayed visually. This visualization gives a clear picture of the evolving gene value at different locations. User intervention is the result of proposing a new solution. This new individuals become a part of the search space in the next generation. The proposed method does not depend on continual interaction of the user. The interaction may be done in any generation and any number of times until a satisfactory or optimized solution is obtained. The fitness is calculated using an objective function and the GA selection method is responsible for selecting the fitter solution for the next generation.

In the proposed work, the visualization technique for the proposed approach is a 2-D graph. The gene values of the search space are represented on this 2-D graph, where the genes having same values are overlapping at the same location. Furthermore, based on the system configuration (system screen resolution and computational speed) used for developing VIGA-2D, maximum chromosome length is 30 for all experimental results. The developed approach also proves the flexibility of producing optimized solution for generating branching structures. For this purpose, the parameters for the Parametric L-System rules are derived from VIGA-2D. However, there should be a little knowledge available for the L-System to run the application for evolving symbols and parameters.

1.7 Organization of Thesis

The remainder of this thesis is divided into four chapters. Chapter 2 contains a review of the literature relevant to the present study and system. This review begins with the discussion of some of the root problems concerning the IGA dependence on human evaluation and selection. The next relevant topic included in this review is the discussion on existing approaches for projecting multidimensional data on lower projection. The following section deals with the various existing approaches for human-machine interaction. Finally, a discussion on application of GA for L-System concludes this review. Different classes of L-Systems are also discussed in this chapter. Most importantly, this chapter discusses the limitations of the existing approaches that motivate the proposed research.

Chapter 3 explains the proposed methodology and discusses its components. The explanation covers the proposed visualization technique in detail. It also discusses the technique used for interaction and the impact of that interaction on the population. This chapter also discusses the feasibility for generating the branching structures using the proposed approach.

Chapter 4 discusses the performance of the proposed approach by comparing its results with the Simple Genetic Algorithm. Evaluation is carried out based on human interaction, fitness and convergence rate. It also describes the results achieved by applying the proposed approach for generating branching structures.

Chapter 5 concludes this thesis by providing a summary of the work. This chapter also summarises the contributions made in the thesis and presents the future directions that can be further taken based on this work.

CHAPTER 2

REVIEW OF RELATED WORK

2.1 Chapter Overview

The Genetic Algorithm (GA) is an efficient search method that is widely used to generate a variety of solutions or an optimized solution [9]. Interactive Genetic Algorithm (IGA) uses the same methods and operators as GA does, except for the fitness evaluation and selection. The purpose of this chapter is to survey the existing techniques and discuss important contributions in the field of IGA. Furthermore, this chapter also survey the existing techniques for visualization of multidimensional data on lower projection. These existing techniques are discussed to establish a background context for the visualization technique developed in this thesis. By describing the current literature, this chapter intends to highlight limitations of the existing approaches.

A detailed survey of the existing IGA techniques is discussed in Section 2.2. Section 2.3 discusses existing techniques to project high dimensional GA data onto a lower projection. Visualization of branching structures and the L-System with its theoretical background and different classes is discussed in Section 2.4. This section also provides reviews of different existing techniques to optimize the symbols or parameters of the L-System with GA. Section 2.5 concludes this chapter by summarizing existing and recent work in IGA.

2.2 Interactive Genetic Algorithm

The origin of IGA is from 1989, when Interactive Evolutionary Algorithms (IEA) were first demonstrated by Dawkins [27] to create a visualization tool to model an artwork called bimorphs. Takagi [28], in his survey, reported categories of Interactive Evolutionary Computation with two definitions, i.e. narrow and broad definition. According to the narrow definition, the human evaluation is used as the fitness value for an optimized solution. Some of these applications, and their advantages and disadvantages are discussed in detail in Section 2.2.1.

According to the broad definition, the human-machine interface is used to solve different problems using GA. In human-machine interactions, the applications are based on a user's preference and selection or it may use some other AI techniques, i.e. a classifier or fuzzy logic to approximate fitness values. Takagi and his fellows have reported many advantages and limitations of using interactive technique for GA in several research works [15, 22, 25, 41, 42]. However, it was noted that most of these existing works were subjects of continues involvement of user [15, 22, 41, 42]. The detail literature review for these applications is discussed in Section 2.2.2. This section concludes with the application of IGA in various fields of science. Section 2.2.3 will do a review on existing techniques with dynamic population size. Different existing application with IGA has been discussed in the Section 2.2.4.

2.2.1 Human Based Selection and Evaluation.

The traditional way of using IGA is to assign fitness and to select the parents for the next generation with human interaction. In this way, this GA visualization gives a suitable solution for the problem in which inference of the user is necessary to have an opinion for the evaluation and selection of solutions for next generation [29]. In another sense, human intuition and emotion accordingly, are needed to complete the evolution process. Another advantage of these existing applications is to help the user to draw or select the individuals according to a visual picture of that object in his mind. These applications are mostly used for model representation of individuals. These techniques are applicable to the problems in which computational time is not a critical issue [15]. A variety of solutions can be obtained by exploring the search space. Using these techniques, human selection can bring an optimal solution, in fewer numbers of generations with a smaller population size. This technique has been successfully applied to 3-D modelling of different artistic applications, for example, fashion design [14], 3D geometric shapes [30] and modelling of 3-D flowers [21]. These applications are based on the phenotype or genotype representation of a problem.

In the visualization of 3D geometric shapes [30] and fashion design [14], the evaluation and user selection judge the aesthetic quality of the model to be selected as parents for the next generation. The individuals of the current generation are displayed in genotype form on the screen as a 3D model. In [21], a similar approach is adopted for phenotype representation of 3D flowers with a Structured Directed Graph. In their approach, each schema (gene) is represented as a graphical shape; in this case they used a total of 10 schemas. The next generation is evolved according to user perception, whereas in the last population they have selected a random solution to draw a flower.

In [31], a 3-D graphic model has been created for manufacturing layout designs. Two different phases are used to run GA. During the first phase all individuals are evaluated by the user. Whereas for the second phase, fitness function is used to assign fitness. During the search process, the user may switch to any phase at any time. Another interesting application is developed by [32], for ubiquitous 3-D graphic models using mobile devices. The evaluation and selection of models are done by the user with a mobile device.

The drawback of these existing approaches lies in their complete dependence on the user. On the other hand, assigning fitness to each individual in search space create a tiresome environment for the user.

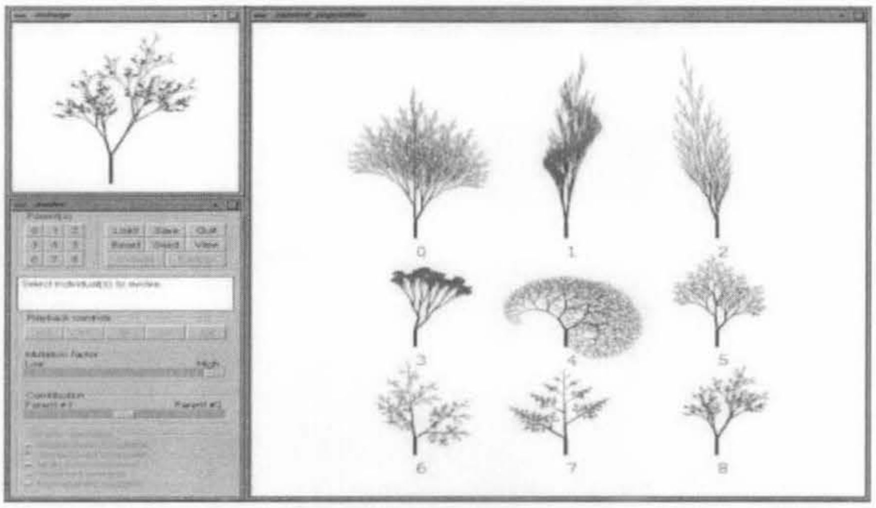


Figure 2.1:A User Interface for Selecting Trees for Recombination and Mutation [33]

The TIGA technique is also used to do the breeding and recombination for the next generation [29, 33]. A user interface is created which allows the user to selects the models, can manipulate its parameters, and selects the parents for the next generation for breeding. The work done in [33] involved a model visualization of trees (see Figure 2.1). These trees are generated using the L-System. The user involvement is to do the selection of parents for recombination and mutation. In this way, the generated trees are according to user perception.

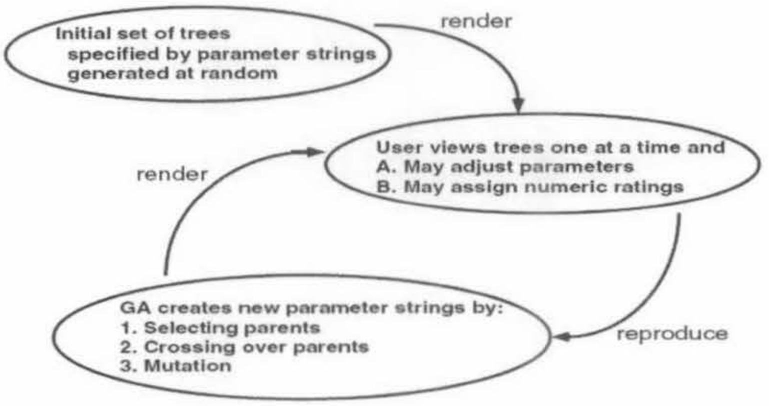


Figure 2.2: Working Mechanism of GENTree [29]

A procedural 3-D model of trees [29] (see Figure 2.2) is generated using IGA. Evolution starts with the initial population, generated by random parameters. Later on the user may adjust parameters to draw a tree. The user can also give a rating to this tree, considered the fitness value. During this evaluation, if the user misses some parameters for the adjustment, then it can be done by GA. Moreover, the user rating of trees is not necessary in every generation. The main focus of their work is not on creating a virtual scene, but rather on giving an idea of how to adjust the parameters interactively. The parameters of two different trees are crossover to produce two new sets of parameters (offspring), which are again displayed onto the screen for user evaluation. Adjustment of the parameters is well addressed in their work; the user may select the trees for breeding, and the user may or may not give the fitness evaluation for the generated trees.

The IGA technique is also used to model virtual scenes [34, 35]. For these applications, the searching of GA also depends on the user perception. The interface window in [34] (see Figure 2.3) represents four virtual scenes developed by 4 elements namely: terrain surface, clouds, trees and sky. The user interface helps to selects the best scene for the next generation. In this application, it is not necessary to assign a fitness ranking to each scene. Instead, the user selection is based on an element (terrain surface, clouds, trees and sky). The tree generated for this system was created by using the parametric context-free L-System. In this way, they have contributed to making random virtual scenes for a forest by using TIGA.

IGA is used for creating visual scenes for generating a software robot [35]. The genetic representation is based on homologous chromosomes. User involvement is required in each generation that produces fitter solutions for the next generation.

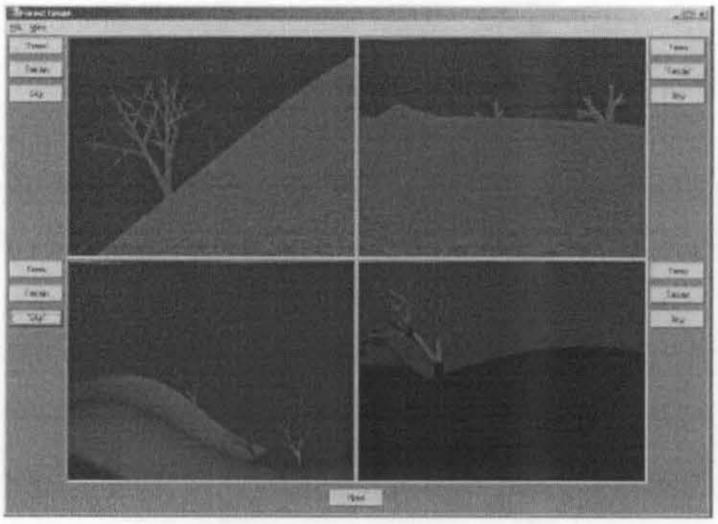


Figure 2.3: User Interface for Selecting Virtual Scenes [34]

Numerous works have been done in previous years to present the evolution process using a graphical user interface. All of these systems work with a model of individuals. Up to now, IGA has been applied to solve problems in several fields. These existing applications are based on addressing both optimization problems and the selection of variety of solution for a problem. It has especially enabled production of very attractive solutions for artistic problems such as 3-D CG lighting design support [22] , animal and plant evaluation using IEC [36], interactive design for websites [37], traditional or fashion designing [14],[38], [39], 3D modelling for geometrical shapes [40], and optimizing image enhancement filters [41]. A further survey of using IGA for evaluation and user interaction may be found in a survey report by Takagi in [42].

Table 2.1 shows some selected existing application based on TIGA. Drawbacks of all above discussed applications are that they need the user involvement in each generation, which create a tiresome environment for user. Due to model representation, the user cannot explore the internal structure and parameters of chromosomes (gene values). Furthermore, the interaction in every generation becomes infeasible for the optimization problems having large search space.

Table 2.1: An Overview of Existing Applications with TIGA

Research Work	**Human Rating & Evaluation**	**User Perception**	**Optimized solution**	**Small Population size**
H. Nishinoet al.[30]	√		√	√
S.B. Cho, et al. [14]	√	√		√
H.J. Min,et al. [21]	√	√		√
R. Curry[33]	√	√		√
A. M. Brintrup, et al.[31]	√		√	
Bruce Merry, et al. [34]	√	√		√
K. Aoki,et al.[22]	√	√		
H.S. Kim, et al.[39]	√	√		√
N. Hiroaki, et al [16]	√	√	√	
H. Nishino, *et al* [32]	√	√	√	
G - Yi-nan at al [43]	√	√	√	√

2.2.2 Human-Machine Interaction

In TIGA, the role of user and GA are separated, i.e. the user does the selection and evaluation for individuals and GA performs the search [15]. However, it often creates fatigue and a tedious environment for the user. Since user fatigue is a main problem in IGA, therefore, researchers are taking keen interest in alleviating user fatigue. Several approaches have been proposed to solve this problem and to improve the GA searching ability towards fitter solution with IGA technique [17, 25, 44].

A human- machine interaction is introduced to create an interval level between user and system to produce a fitter value. Using this technique, a discrete fitness value is introduced [25] to evaluate the solutions for the next generation. They proposed to assign same fitness value to all individuals having similar features. In this way, they reduce the user fatigue for evaluating and assigning fitness to each individual in the search space. Some approximation approaches have also been used to solve the problem for assigning fitness. A Neural Network has been adopted [44] for assigning fitness values after learning the human selection behaviour for different individuals during the evolutionary process. Feed forward Neural Network is tried [45] for predicting human evaluation and displays the individuals in decreasing arrangement. However, in their work they reported that predications given by the Neural Network were less accurate than fitness function for assigning fitness.

A Model based visualization is proposed [26] in which the fitness of an individual is not assigned by subjective or objective function. The fitness is calculated on the base of time spent by a user to make a solution which is satisfied or not satisfied. In this way, the difference of this time while evaluating the solution according to user satisfaction is considered as the fitness value of the individual. The selection of parents for the next generation is done by GA.

A multivariable problem [17] is addressed to solve the fitness evaluation problem using the Neural Network termed as General Regression Neural Network (GRNN). GRNN approximates the aesthetic intention of the desirable solution by the user with its learning mechanism, whereas IGA is used to evolve the next generation. In this way, they save the user from the tiresome work of selecting and assigning fitness to the best individual. They applied their proposed approach to designing the cartoon style faces on coffee mugs (see Figure 2.4). These designs are displayed on a grid window for selection by the user. Figure 2.5 shows the general flow of their system.

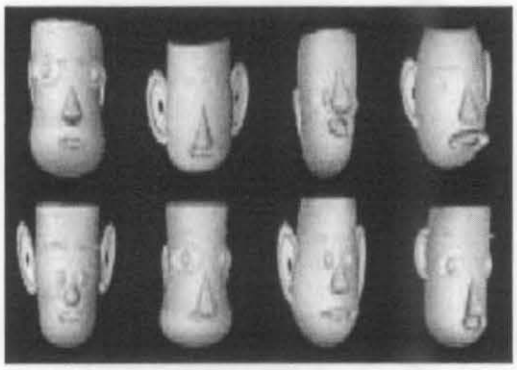


Figure 2.4:Cartoon Faces Generated using GRNN [17]

This GRNN application works in two phases. The first phase starts with a random population. Then, it evolves generation to generation with the artificial selection of designs using IGA. The next generation is derived using asexual crossover and with a high rate of mutation. The user may select one or many designs in one generation. The selection and deselection of the user history are saved, analysed and formulated in the Neural Network memory. After several interactions and generations, the second phase is started. The Neural Network is used to approximate implicit mapping between the evolved process and the user responses. The user responses are analysed and feedback is given to SGA for automatic convergence.

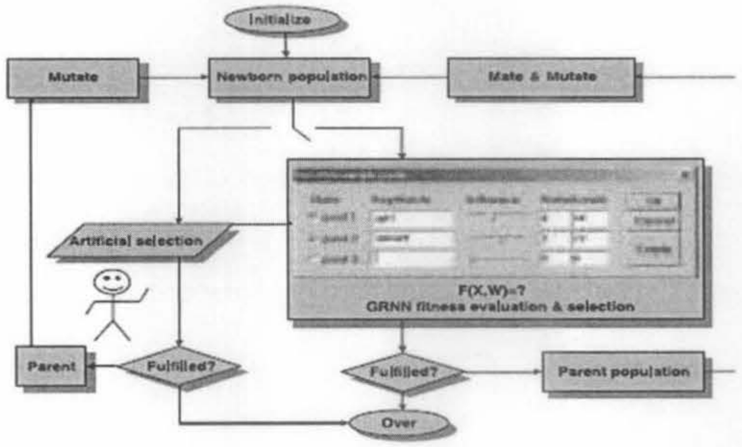


Figure 2.5: General Flow of System [17]

The literature survey shows that, most of the applications developed with the human-machine interface are also based on the user selection or evaluation. Although the fitness is not evaluated by the user, instead; fitness approximation misleads the gradual and fuzzy evolutionary process and restricts it under some fixed parameters. For example, the fitness values are discrete in nature and user select any best option from them [15] or some probability values are taken to evaluate the individuals of each generation. Furthermore, there has been no countable work done in these applications to improve the searching performance of GA using visualization techniques.

Another idea is to introduce an approach in which user interaction may be involved after a few generations or in certain (pre-defined) generations. The advantage of this approach is that it saves the user from the tedious work of selection and evaluation in each generation. The objective function is used to assign fitness to the individuals; hence, a large search space can be used for finding the solutions. An occasional user intervention is introduced to correct the fitness of individuals used for multi-objective optimization [24]. They proposed this approach to solved problems with the large search space. They employed this technique to optimize the parameters for aircraft design. The user interaction was after certain number of generations. Objective function is used to calculate fitness. Figure 2.6 shows their designed system. In their work for representing multidimensional data having different objectives and constraints, different graphs are used. In total, they have 35 design variables separated into 5 groups. The size of the population is 100 - 150.

The drawback of their system is in using different graphs for each group. In this way, the overall performance of the system becomes difficult to observe. On the other hand, this technique is helpful to elaborate the usefulness of user interaction to pre-determined generations. In this way, selection or changes in searching becomes easier as it is concerned only with particular generations.

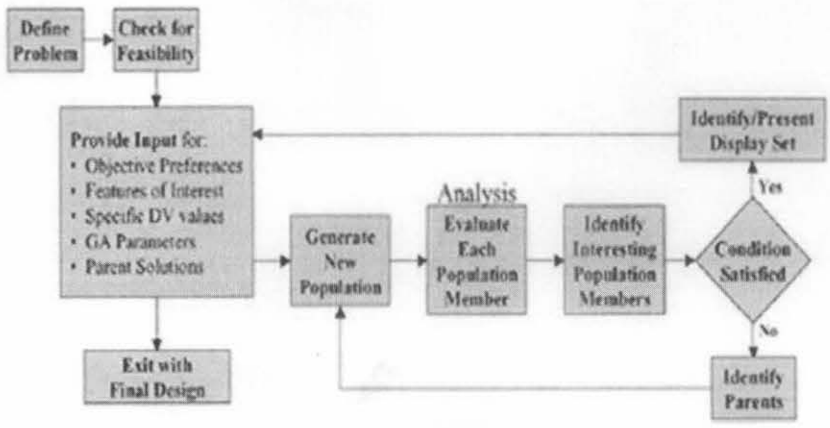


Figure 2.6: An Aircraft Design using IGA [24]

2.2.3 Dynamic Population Size

Adjustment or variation in population size may be observed in natural environments and ecosystems with the change in the number of species. This change depends on the availability of natural resources and the capacity in the ecosystem. Hence, population size is a flexible parameter in natural systems. Initially, in evolutionary computation, this parameter remains a constant over the run; scientists pay more attention towards dynamic values for crossover and mutation operators.

However, a theoretical analysis has been done by [46] and described methods for an optimal size of a population. Several other researchers have also been conducted to address the size of population in different perspectives [47, 48]. In these works, the population size parameter is considered as a flexible parameter and different experiments have been done to change the population size during the GA search process. In most of the previous works, the focus was to control the size of the population with various approaches. Increment or decrement of population size depends on some other factors, i.e. fitness rating and threshold value.

Variation in population size is feasible to evolve engineering problems dealing with a large search space. Some experiments have also been done with the variation in population size with IGA. A successful work has done in [38] , in which they used IGA for evaluating fashion design models with variations in population size. They divided their system into two phases, i.e. the Fluctuant and the Stable phase. In the fluctuant phase, they used a clustering method, in which all the similar individuals work in the form of group or cluster. The population size is large in this phase. The user evaluates the centre of the cluster, and the fitness of all other neighbouring individuals is calculated on the basis of the centre's fitness. The size of the population and the similarity threshold are constant in this phase. In the stable phase, the similarity threshold is varies with the evolution, thus improving the clustering of the population. They adopted variant population sizes in this phase. The increases or decreases in population size depend on the similarity threshold. Some elitism individuals are reserved in this phase, which help to generate improved offspring for the next generation.

2.2.4 Applications of IGA

Applications on GA have extended from computer science to other fields, i.e. engineering [24], physics, social science, economics, molecular biology , biomedical engineering [49] and many other fields [9,50]. On the other hand, IGA is suitable for problems which cannot be easily solved easily by conventional GA, especially where the interference of the user may change the general behaviour of GA, i.e. in aesthetic and artistic problems [14, 51], in image processing applications [41, 52, 53], the travelling salesman's problem [54], chemical optimization problem, engineering design problems [24, 55], and modelling of artificial scenes or trees [56]. All of these above mentioned applications are developed using different techniques of IGA to get optimized solutions. IGA is suitable for applications where the objective function is unable to the assign fitness value or if it is useless. This technique also serves as a useful tool for understanding complex architectural designs and civil engineering applications. One of the main advantages of this method is that it has the potential to obtain solutions according to the user's desire, and to produce many variations in obtained solutions.

2.3 Multidimensional Data on Lower Dimension

Visualization of multidimensional data on lower dimension is used for comprehensive representation for scientific results, and their interpretation or validation [65, 066]. Besides the modelling and visualization of scientific data, these techniques are also used to visualize the hidden process of algorithms including the biological inspired algorithms such as GA [15]. The visualization of GA searching data is based on different techniques, used to transform multidimensional data to one or two dimensions, i.e. Principal Component Analysis, Biplots [57], Distance Maps [58], Sammon Mapping [59, 60], Coverage Maps [61], Distance Maps [61], State Space Matrices [62], SOM [15], and Correlation Tours and Grand Tours [63]. Beside these techniques, Tom Routen [64] proposed a distance distribution histogram for calculating distance between chromosomes for population. This technique is only helpful to determine the frequency of chromosomes. There is no information found for the distribution of chromosome in the search space found using this technique.

Most of these existing techniques do not have common mapping for all generations. As a result, there is a lack of consistent relationships between the plotted points for subsequent generations. The mathematical complexity of these techniques also makes them impractical to apply for real problems.

Based on the techniques involved in the visualization, it is broadly divided into two main categories: (a) Information visualization and (b) Scientific Visualization. Information Visualization (InfoVis) includes visual representation of non-numerical data [65, 66]. InfoVis represents any technique for displaying abstract data and helping to view a large amount of data at once [67]. In other words, InfoVis helps to analyze and understand data in a better way. Different data mining techniques are represented using InfoVis [68, 69]. In this way, business applications are the main category to which InfoVis is applied. Moreover, Web based information [70] is a new trend used to share information through information visualization. Given this idea, the Web becomes the largest source of information. Moreover, financial data and report are also visualized using the information visualization technique [71].

Scientific Visualization (SciVis) includes modelling, representation or simulation of scientific data. This visualization helps to understand the data derived from numerical calculations or from any scientific experiment. SciVis covers a large number of scientific fields, for example simulations of physics based models or chemical processes [72], mathematical modelling [73], and virtual reality applications [74], and biomedical problem based modelling i.e. 3-D modelling for the brain [75], protein structure [76], and MRI [58]. However, visualization of complicated and multidimensional data of GA may also explore using different techniques that covers in SciVis.

For example, in [15], a 2-D map visualization technique is used for GA for improving the searching ability with user interventions. For visualization of GA, they used SOM to map the individuals from n-Dimensional space onto a 2-D space. On a 2-D space, the individuals are represented with different colour intensities. For example, an individual with a higher fitness having has a dark color as compared to an individual with a lower fitness, or new selected individuals are represented in different color.

In their work, the participation of humans is to select the parents for the next generation. For the current search space, a user guesses for the possible fitter individual and sends it to the GA as a parent for the next generation. GA replaces this proposed fitter individual with the lowest fitness individual in the search space. By using SOM during this mapping, all the individuals of n-D mapping become neighbors on the 2-D space. In this way, the new selected individual in the 2-D space takes closest place to the global optimum in n-D space due to a topological relationship. The GA converged to the next generation with the newly added individual.

During visualization in [15], the best solution of the current search space is displayed in a different color, which makes it easy for a user to select the nearest fitter individual. In their proposed approach, it is not clear how they are calculating the fitness for individuals. Furthermore, 2-D mapping is based on all previous generations; for example, if there are 40 individuals, it means 20 individuals x 2 generations. For this reason, SOM is retained in every generation to keep the possible points in 2-D space as 200 x 200. Users need to select the 3 individuals among which the fitter individual will be the parent for the next generation. Another problem with the SOM is that it cannot keep the absolute distance difference in 2-D space, so only estimated distance is mapped to the n-D space again. Furthermore, due to interaction in each generation, fewer numbers of generations are evolved.

Several computer graphic techniques are used to project the high-dimensional data on one or two dimensions, i.e. scatter plots, parallel coordinates or different colour schemes [19]. These visualization techniques give both phenotype and genotype representations of GA data, which is carried out in the form of chromosome or gene values. A pseudo colour strategy is adopted in [19] to display all the individuals of the current generation onto one screen(see Figure 2.7). The brightness of individuals change with the fitness value while with the objective value, the hue changes. This approach is applied to solve the knap sack problem. The chromosome encoding is binary; hence, two colours, blue and red, are used to distinguish between gene values 0 and 1 respectively. The user views the individuals of each generation visually and decides the termination condition. Since the colour scheme is applicable to individuals having binary coding, therefore, it is not practical to apply this scheme on any other genetic encoding.

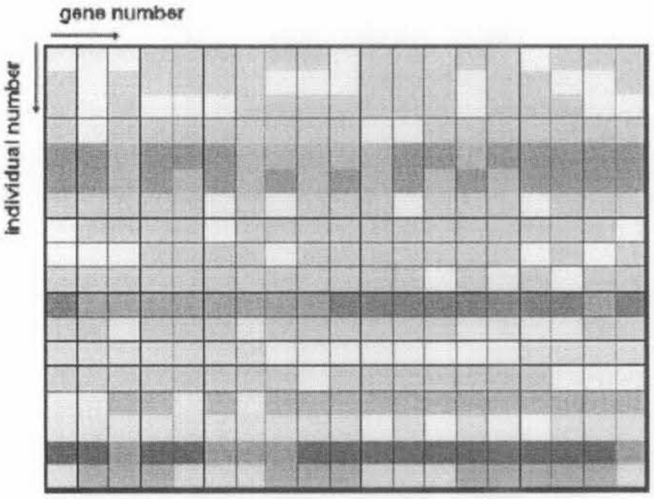


Figure 2.7: Brightness and Hue in Current Generation [19]

A Target Line [18] is projected onto one space for high dimensional data, where all chromosomes are a point on target line as shown in Figure 2.8. The size of the point shows the number of chromosomes projected on it, hence, the larger the size the more chromosomes on it. The colour of the target line indicates the number of points on that part of the line which works as a bar; hence, the darker the line more points projected on it. With the selection of any part of bar, a scale shows the percentage of the population in that area. The user may also change the position of the target line in the search space. However, this change will not inform that what position of the target line could be optimal for visualization of a particular run.

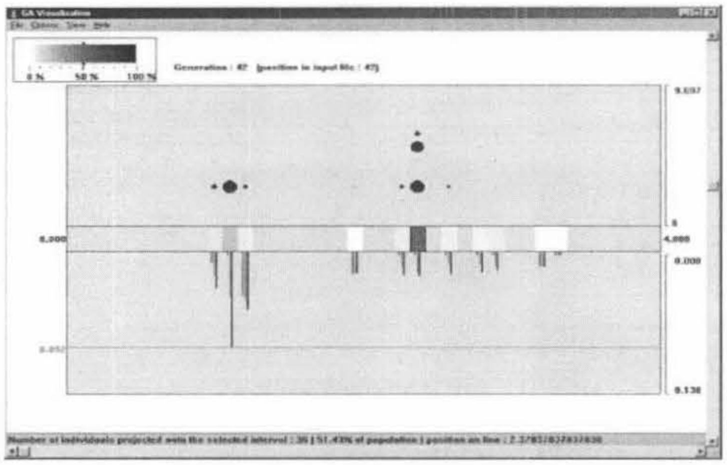
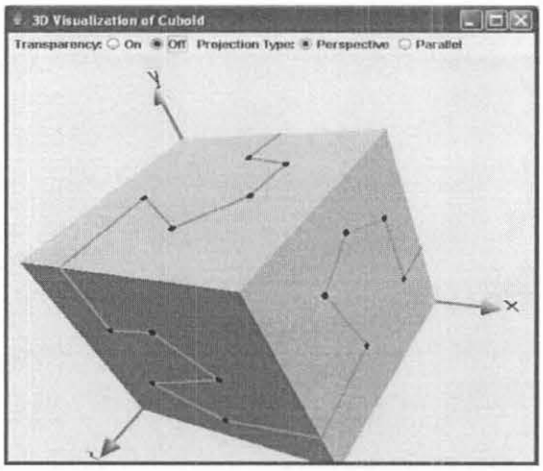


Figure 2.8: Projecting High Dimensional Data onto Lower Projection [18]

Computer graphic techniques are also involved with the visualization of the smallest element of GA known as gene values or allele. Visualization of gene values gives a comprehensive overview of the complicated structure of a chromosome [19]. This technique is used to observe what area of the search space is explored by GA. A travelling sales person [54] is a well known problem in which optimized solution for paths to the cities is obtained. By projecting this data in 3 dimensions, the IGA with user interaction is a suitable idea to solve this problem as shown in Figure 2.9.



3D Visualization of Cuboid

Transparency: On [✓ Off] Projection Type: [✓ Perspective] Parallel

Figure 2.9: Travels Sales Man Problem on 3-D cuboids [54]

2.4 Visualization of Branching Structures

Computer graphic techniques are also used to interpret and visualize the branching structures. These techniques are used to create realistic models for branching structures. In this way, natural looking scenes are generated using these techniques. In branching structures there are different functional modules. These modules work and are arranged together as functional modules to make a branching structure. The geometrical structure of each functional module is similar to the whole. Hence, each module has sub modules and these sub modules have more sub modules. By increasing the levels of the modules, the branching structure becomes complicated.

The natural patterns applied by nature for these branching structures can be visualized using the L-System. The L-System is a rule based system used to interpret these repeated structures. This rule based system start from a simple form, i.e. axiom and move towards a complex structure [77]. The turtle graphics are used to build a geometrical interpretation of L-System strings. The following section discusses the functionality of the L-System and its classes. In Section 2.4.2, there is a brief overview on the Parametric L-System. Existing work for using GA for evolving symbols or parameters of the L-System are discussed in Section 2.4.3. This section concludes with the discussion on different applications of L-System.

2.4.1 Functionality of L-System

The L-System is a mathematical formalism used widely for modelling and visualization plants and branching structures with computer graphic techniques. Lindenmayer [78] was a biologist who proposed the L-System for the first time in 1968.He used a rewriting mechanism for generating cell division in multi-cellular organisms. Later on he also used it for modelling plant growth. But Honda [79] was the first scientist who introduced plant modelling. The L-System works in the same way as the Chomsky grammar does but there is a difference in the method of applying productions rules, thus making L-Systems different from all other rewriting mechanisms. The rules of the L-system are applied in parallel [80], simultaneously replacing all the letters in a string in one step. While in Chomsky grammar the rules are applied in sequence, one string is replaced in each step. The parallel replacement also makes a difference between procedural modelling and rule based modelling [77]. This property reflects the biological inspiration of the L-system. These rules work recursively to model complex plant and branching structures. The L-System is categorized on the basis of the grammar they use, because different grammars generate different formal languages [81]. This categorization represents different classes of the L-System. These are described below:

- **Context free L-System:** A system in which each production rule refers only to an individual symbol or module and not to its neighbours. The sequence generated by this L-System is self similar at all levels.

- **Context Sensitive L-System:** In the context sensitive L-systems [82, 83], the production rule depends not only on a single symbol but also on its neighbours. These L-Systems are used to add environmental parameters, i.e. weather, clouds and gravity [34].

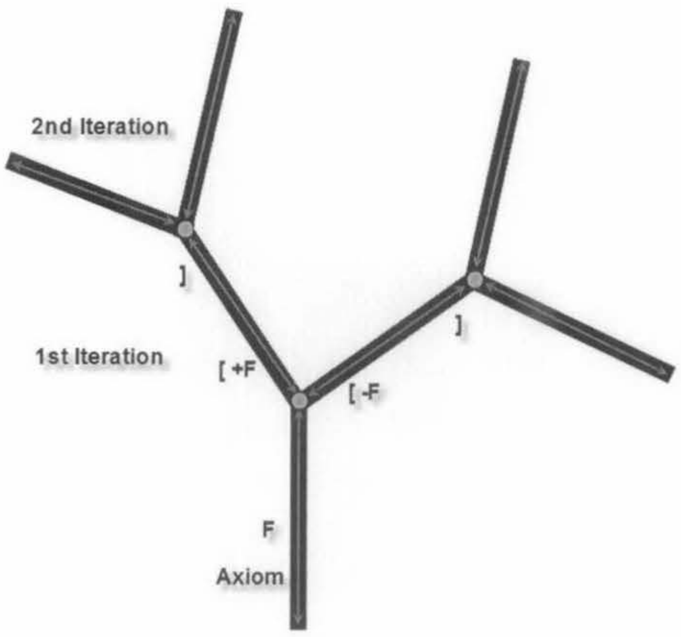
- **Deterministic L-System:** Deterministic L systems always produce the same development sequence. In other words, when there is only one rule for all levels of iteration.

- **Non Deterministic L-System:** If more than one successor is used to create production rules than it is known as a non-deterministic L-System. During derivation, at least one symbol or module should have more than one production rule for these L-Systems.

- **Bracketed L-System:** The Bracketed L-System is an extension to the L-System for generating tree like structures. The concept of using branching symbols with the L-System was introduced by [78]. To represent branching structures in the turtle graphics [79] , two symbols are used:

  [= push the current state of turtle onto the stack.

  ] = pop a state from the stack and make it the current state of turtle.



Axiom = F

F ->F[+F][+F][-F[-F][+F]][+F[-F][+F]]

An example for [push, pop] in/out from stack with $\alpha = 45.95$ for all angles.

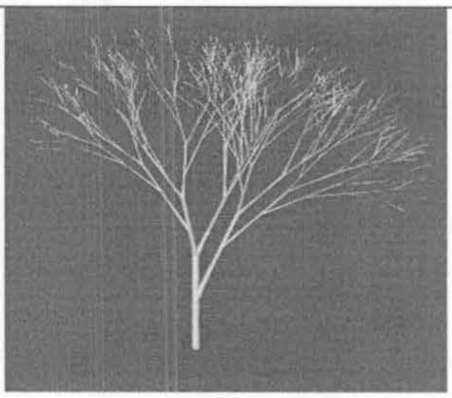
These classes of the L-System depend on the way of determining the production rules thus, they vary from each other. Different classes may be combined together to produce more attractive results. The developed rules may be deterministic, i.e. context free which are known as the simplest form of the L-System, and work with symbols, or they may be context sensitive, i.e. different parameters or probability values are used with the rules. The Parametric L-System is a new form of the L-System, is used to model the growth process of a tree or plant by defining rules. In the Parametric L-System random values assigned to the symbols, to create more natural looking models.

2.4.2 Parametric L-System

The simple form of the L-System was unable to model the growth process of plants due to its discrete nature. Therefore, the Parametric L-System was introduced; it is a further extension of the L-System and is used to visualize the growth process of plants. In Parametric L-System, numerical parameters are associated with the symbols. These parameters control the effect of production rules during iterations. In other words, the idea behind the Parametric L-System is to use parallel rewriting with parametric words instead of strings of symbols [82]. This mechanism brings more variety of branching structures by assigning dynamic values of parameters during expansion.

In recent years, the Parametric L-System has been widely used for modelling and visualizing the growth steps for tree and plant structure. They may be called as development models which generate beautiful, smooth, fast growing animations for branching structures. Some main features of using the Parametric L-System are [82]:

1. Instead of multiple discrete units, the Parametric L-System expresses a wide range of angles and their length.
2. It uses arithmetic expressions, especially for demonstrating the growth process of plants.
3. The presence of numerical parameters makes it easier to change the structure by only interacting with / changing the numerical values.
4. The Parametric L-System has the ability to control iteration from one step to the next, resulting in a smooth visualization.

Parametric L-System Rule	Generated Tree Structure with Rule
Constants: Ang01 = 435.74, Ang02 = 832.63, rot = 20.95 , width = 1.932 Axiom: !(1) F (200) / (15) A Rule 01: A -> !(width) F(50) [-(rot) F(150) A] /(Ang01) F (150) [& (rot) F (50) A] / (Ang02) [&(rot) F(50) A] Rule 02: !(w) -> !(w*width)	

Example: Tree generated after 05 Iterations using a Deterministic Parametric L-System.

2.4.3 Genetic Algorithm for L-System

The L-System is widely used to generate natural looking scenes, fractals or branching structures using computer graphic techniques. However, to construct the rules for the L-System is time consuming, and usually these resulting rules are not dynamic in nature. Therefore, the researchers use GA to derive L-System rules because of its random search ability. This ability of GA helps to derive a number of variant rules. In [84] trees structures are generated randomly by GA and compared with a database of target tree structures. The symbolic encoding is used; therefore, to make every individual meaningful, a repair mechanism is used to improve the symbolic structures and to prevent synthetically incorrect results. This repair mechanism is applied on the individuals of the population before calculating fitness. The limitation of their works is in using a small search space. They have used 2 rules for generating tree like structures and modelling them onto the 2-D space. Therefore, there system has failed to cover the complexity of rules and trees.

L-System rules for different classes may be used together to be optimize using GA. For the most part, these systems work with the same phenotype of branching structures, and GA is used to optimize the rules for the L-System. A system named *lworld* has been developed [85] consisting of modelling different classes of the L-System, i.e. Parametric, Timed, Stochastic, Bracketed and Real Time. The main contribution is to give a facility to the designer, who may manipulate the fitness values, and can change sub-population and other settings for the L-System, i.e. probability values for the stochastic L-System.

Besides the symbols, the parameters can also be randomly generated. A 'sketch and grow' interface is developed to retrieve an L-System string from user sketch [86]. The stroke input is translated into L-System symbols and parameters, which indicates the height, main axis and a number of iterations required until which the tree should be grown. The rules of L-System are developed with the help of user input that is used to model the further growth of tree according to the number of iterations. For deriving the parameters closest to the input of user, GA is used to optimize the parameters. The rules derived according to optimized solution are used to generate branching structures using turtle interpretation onto 3-D space.

Most of the researchers used Evolutionary Strategies (ES) for evolving parameters having encoding in the form of vectors of real values [49, 87]. System *GREADEA* [49] has been developed in which combined evolutionary operators are used, i.e. GA is used to derive symbols and an evolution strategy is used to evolve parameters. This system describes the structure of the human retina using L-System formalism. They have created a database of images for the Parametric DOL-System by applying several image processing algorithm techniques on images. However, if images which are developed by a scanner laser ophthalmoscope *(SLO)* are corrupted or incomplete, then their defined technique is not able to overcome this problem.

A parallel evolution approach is adopted by [87].The symbols and numerical parameters of the Parametric DOL-System are evolved by evolutionary algorithms, in which symbols are optimized by GA and parameters by using evolution strategies. The idea behind their research is to run more than one population in parallel having the same size. There parameters are independent from each other in the search space. These populations can exchange the best individuals with each other during the evolution process. The user interaction only involves selecting the number of populations to be evolved and the user could change the mutation, crossover rates and desired generation numbers. A simulation of 2-D plant morphology is developed in [88]. The variable length (genotype) is based on the given L-System rule. The system is divided into two types of selections: (a) User based interactions to achieve desired solutions. (b) Automatic evolution is carried out using GA. A bilateral fitness function is used to evaluate the fitness of solutions.

Besides of using evolutionary strategies, a tag function concept is introduced by [89] as a replacement of real value parameters. This approach is applied onto the modelling of a Leaf. The L-System is used to construct the shape of the leaf by rewriting rules, and tag functions are evolved using GA. The purpose of using the tag function is to reduce the time for deriving rule in every step, because these functions replace derivations which have been done before. The user only requires changing these function values instead of deriving an axiom again. However, they have created only the leaf shapes using their proposed approach.

An interactive approach of GA has also been used successfully to generate tree like structures. The Parametric L-System and GA are used to develop an environment [33] for the modelling of plants with user involvement. This interaction is based on the plants of 9 typical types. The crossover is done between two models chosen by user, and mutation operators are applied on one the selected plant models; this process continue until obtained the solution. The disadvantage of this application is that the whole system works with the visual models, and the complexity of the rules in the form of gene values remain hidden from the user's eye.

Table 2.2: An Overview of Existing Work for Visualization of Branching Structures with L-System and GA.

Authors	Symbols Evolved by GA	Parameters Evolved by GA	Interactive Visualization	Optimized Solution
Bian Runqiang, et al. [84]	√			√
Hansrudi Noser, et al.[84]		√		√
R. Curry [33]	√	√	√	
Gabriella Koka, et al. [49]	√			√
Kokai, G, et al.[87]	√			
Gabriela Ochoa [88]	√		√	√
Yodthong Rodkaewl, et al. [89]		√		√
L E. Da Costa,et al.[0090]	√	√		√
A. Daniel et al [0091]		√		√
N. Zakaria [86]		√		√

Table 2.2 shows a list of some selected existing approaches for visualization of plants or tress using L-System and GA. All these system give a good overview for all the important classes of the L-System but the interaction does not directly play a part in changing the evolution process. All of the above existing discussed systems are

well address but only to their specific problem. Additionally, all of these existing systems are restricted to the static evolution process of GA. One step forward is to use the interactive approach of GA to derive the rules for the L-System.

2.4.4 Application of the L-System

The L-System works as a powerful tool to generate complicated structures in a 2 or 3D environment with a small number of rules. Besides modelling and the visualizing of plants and trees, the L-System is also used in other applications. For example design patterns [92], music rhythm [93], draw sculptures or artistic drawings [95, 96] or generate virtual creatures [96]. The context-sensitive L-System is a powerful rule based system for generating complex structures with environmental variables [97, 98]. The stochastic rules of the L-system are applied to generate random structures with the same rules.

The Parametric L-System is the extension of the L-System used to generate and simulate the growth process of plants and trees [84]. It also gives a facility to show the complicated structure of human organs, for example, modelling blood vessels of the retina [99] or a growth simulation of the stomach in an embryo [100]. L-System applications are also widely used to give a virtual environment in games and movies. It has been also used to give special effects in movies like, making visible or invisible the vessels in Hollow Man 2and the neurons in Fight Club [101]. Recently, the L-System was also used to create building infrastructure and road networks [102]. Besides this, an interactive L-System [103, 104] is also used to develop a relationship between natural and artificial environments. Some advantages of using the Parametric L-system are: (a) its ability to use arithmetic expressions in rules, (b) a large variety of angles and expressions to control and demonstrate the growth process or to create virtual scenes, and (c) its ability to control the derivations in each iteration.

2.5 Summary of Chapter

This chapter has presented the review of current literature that has led to highlight the existing techniques for solving problems with IGA. This chapter also provides a survey on existing visualization techniques to improve the performance of GA with an interactive environment and different visualization techniques used for understanding the GA searching mechanism. This chapter has also discussed the limitations of existing techniques in previous sections. In this way, this chapter identifies the problems in existing approaches, thus making a base for the proposed work. From discussing all this literature, it is concluded that the existing techniques for visualization based on individuals and continue human interventions with GA search space are solutions for artistic problems with a small number of generation. However, there is still room for exploring new techniques for the solving problems need large number of generations, to avoid continue user interventions and for the visualization of GA search space based on gene values.

Different current approaches are also explored in this chapter for assigning fitness using the objective function to show that IGA techniques may also use to visualize the large search space. However, it was noticed that in most of the existing techniques there is no any techniques to propose a new individual or new gene values at different gene locations in the current generation to become a part of the next generation. It is also noticed from the existing approaches that the techniques for mapping multidimensional data in lower dimension based on gene values is lacking.

The existing approaches using GA for optimizing or deriving L-system rules are also discussed in this chapter. The Parametric L-System is a class of L-system used to generate branching structures and to model the plant growth. These kinds of visualizations are also used to develop the background scenes for animated games, and virtual reality scenes in movies or in animated cartoon movies. After a thorough examination of relevant literature, it is also noticed that there are only a few existing techniques available to evolve the parameters of the L-System with IGA. The next chapter presents our proposed methodology in detail.

CHAPTER 3

PROPOSED APPROACH

3.1 Chapter Overview

This chapter will describe in detail the procedure and functionality of the proposed approach. This chapter will describe in detail the technique used to bring the multidimensional data of GA search space on 2-Dimension. Secondly, this chapter will address in detail the technique used for accelerating the performance of GA towards optimal solution with user interventions. This user intervention in the proposed approach is different from existing IGA techniques in which interference of human is necessary in every generation for evaluating and assigning fitness to individuals. In proposed approach, user may interact in any generation and this interaction is used for proposing a new individual into search space. Hence, the users are not involved in evaluating and selecting the parents for next generation, which prevent them from fatigue. In proposed approach, a human–machine interaction is used in which fitness is assigned by the objective function and selection of the parents for the next generation is done with the GA. Hence, the proposed methodology works with large numbers of generations as compared to the existing techniques of IGA in which small number of generations have been used to prevent user fatigue. The application of proposed methodology is to derive the Parametric L-System rule for further growth of branching structures. This chapter will also discuss the technique used to derive the rule from best solution of VIGA-2D used for modelling the further growth steps of branching structures.

This chapter is organised as follows: Section 3.2 describes the problem formulation. The proposed approach for interactive visualization of the search space is discussed in Section 3.3. Different GA selection and reproduction methods used in proposed work are discussed in Section 3.4. Section 3.5 discusses an application used in this research which involves the optimization of parameters for a Parametric L-System used to derive the L-System rules for modelling the growth process of branching structures. The chromosome encoding and visual representation of gene values for test functions is discussed in Section 3.6. Section 3.7 provides a summary of the chapter.

3.2 Problem Formulation

The objective of this research is to propose a technique for an interactive visualization of multidimensional (n-D) data of GA on a 2-D space and to accelerate the performance of GA with human intervention. For this reason, the user actively participates in searching by proposing a new individual in any generation. The new individual becomes a part of searching process into next generation. Thus interaction of the user leads to a faster convergence of the search and faster convergence results with less user fatigue. Prior to becoming a part of the search space, the fitness of the new individual is compared with the average fitness of the current generation. Thus the higher the fitness of the proposed individual, the more chance it has to be selected in the next generation. This interaction also affects the population size; each successful interaction will increase the population size. Thus proposing a fitter individual using interaction enables the GA to converge more efficiently in fewer generations.

Evaluation: Let α is the search space for GA. Then $\alpha = \{\beta_1, \beta_2, \beta_3, \ldots\ldots\ldots\beta_k\}$, where β represents generation evolved in search space for k numbers. Each generation β consists of δ chromosome, represented by $\beta = \{\delta_1, \delta_2, \delta_3, \ldots\ldots\ldots\delta_m\}$, where m is the size of population. Each chromosome δ of population consists of gene values denoted as λ makes a set of $\delta = \{\lambda_1, \lambda_2, \lambda_3, \ldots\ldots\ldots\lambda_l\}$, where l is the length of chromosome. Hence the multidimensional data for each generation consists of:

$$\beta = \begin{bmatrix} \delta_1 \\ \delta_2 \\ \vdots \\ \delta_m \end{bmatrix} = \begin{pmatrix} \lambda_{11} & \lambda_{12} \cdots & \lambda_{1l} \\ \lambda_{21} & \lambda_{22} \cdots & \lambda_{2l} \\ \vdots & \vdots & \vdots \\ \lambda_{m1} & \lambda_{m2} \cdots & \lambda_{ml} \end{pmatrix}$$

For visualization, each generation β should be represented on the 2-D graph in such a manner that the x-axis represents each gene location and the y-axis displays each gene value ($\lambda_1, \lambda_2, \ldots\ldots\ldots\ldots \lambda_l$) as shown in Figure 3.1.

In the proposed research, any generation may be interacted by the user. Let the user interact in β_i generation, and propose new gene values, which are a set of $\rho = \{\lambda_1, \lambda_2, \lambda_3, \ldots\ldots\ldots\ldots \lambda_l\}$. These gene values become a part of the next generation as a complete set of δ, where $\delta_{m+1} = \rho$, and ρ represents a new proposed individual. Hence, it depends on the fitness of individual ρ, in order for it to become a part of the next generation or be discarded by the GA process. In this way, the discarded individuals will not be considered as a part of evolution process. The next section will describe the functionality of the proposed approach in detail.

3.3 Background of the Proposed Approach

The visualization of the GA makes understanding of the search space easier. This visualization is based on either displaying the individuals or gene values on lower dimension in order to understand the convergence behaviour [18, 19] or to make this algorithm an interactive technique with the involvement of user [14, 30]. Hence, these techniques are useful for observing the effect of the user interaction during the searching process. A thorough literature survey, in chapter 2, shows that most of the existing IGA application works with a small number of generations [14, 21, 29]. The problem introduced to GA was in the form of models, and the user interaction in each generation used to bring the best solution for next generation. In these existing applications, the fitness assigned with the some range i.e. 1-5, and the user selects the appropriate fitness for the solution. In this way, the selection and evaluation of each generation is based on the user's decision. This continues involvement of users often create a tiresome environment for them.

Existing literature shows that a numerous work has been done to reduce the user fatigue [15]. For example using Neural Network [44], discrete fitness values [25] or by calculating the time spend by a user on a particular individual [26]. However, it was noted that all of these existing techniques found on the base of involvement of user in every generation. It was also observed that IGA existing techniques were based on visualization of individuals as 2 or 3-D models. Thus the user remains unaware during the searching process and with the convergence behaviour of GA. Based on the limitation of existing applications for IGA, an interactive visualization of multidimensional data based on the gene values and to accelerate the performance of GA by proposing a new individual is the main focus of the proposed approach.

3.4 Overview of the Proposed Approach

In this present research, an interactive visualization technique for the multidimensional data of GA on 2-D space is proposed. The proposed technique is used to increase the performance of GA towards fitter solutions with the help of human interventions. Moreover, the proposed approach is also capable for observing and analysing the convergence of GA towards an optimized solution using a graphical interface. Different steps have been taken to accomplish this goal. According to the problem formulation in Section 3.2, a tool named VIGA-2D (Visualization of Genetic Algorithm on 2-D Graph) has been designed and developed (see Appendix A for design and implementation). This visualization tool has been developed to show the clear picture of the searching process of GA in every generation.

Besides the visualization techniques used for representing the GA data, the performance of GA depends on many factors including a good fitness function, population size and choice of GA operators (crossover and mutation). There are four main steps, which are necessary to be taken by GA to complete a cycle [12]:

a) Representation of the problem in the form of a genotype

b) Decide on the fitness function, which is based on the expected GA solution

c) Define the method of reproduction, recombination and mutation

d) Decide on the termination criteria based on the nature of the problem

A complete GA process needs all the above mentioned steps to obtain an optimized result. These GA steps work in the same sequence for IGA. Since the user is involved in IGA to complete the search process, however it needs some modifications in regards to the problem.

All of these steps have been taken in the proposed approach. However in the proposed approach, the role of the user is for proposing a new individual in the search space instead of assigning fitness to the existing individuals. Listed below are some advantages of the proposed work.

- The proposed approach works with a human-machine interaction.
- Instead of the fitness value being assigned by the user, the objective function is used to calculate fitness for each individual.
- The visualization is based on the gene values of each generation.
- Human interaction is not involved with the selection or rating of the solutions; instead the user proposes a new individual in any generation.
- There is no need to keep the generation numbers smaller because the user interaction is not necessary in every generation.
- Any generation may be interacted by the user.
- The working mechanism of GA operators (crossover and mutation) and selection of individuals work as a hidden process.
- The GA search process continues until the user terminates it.

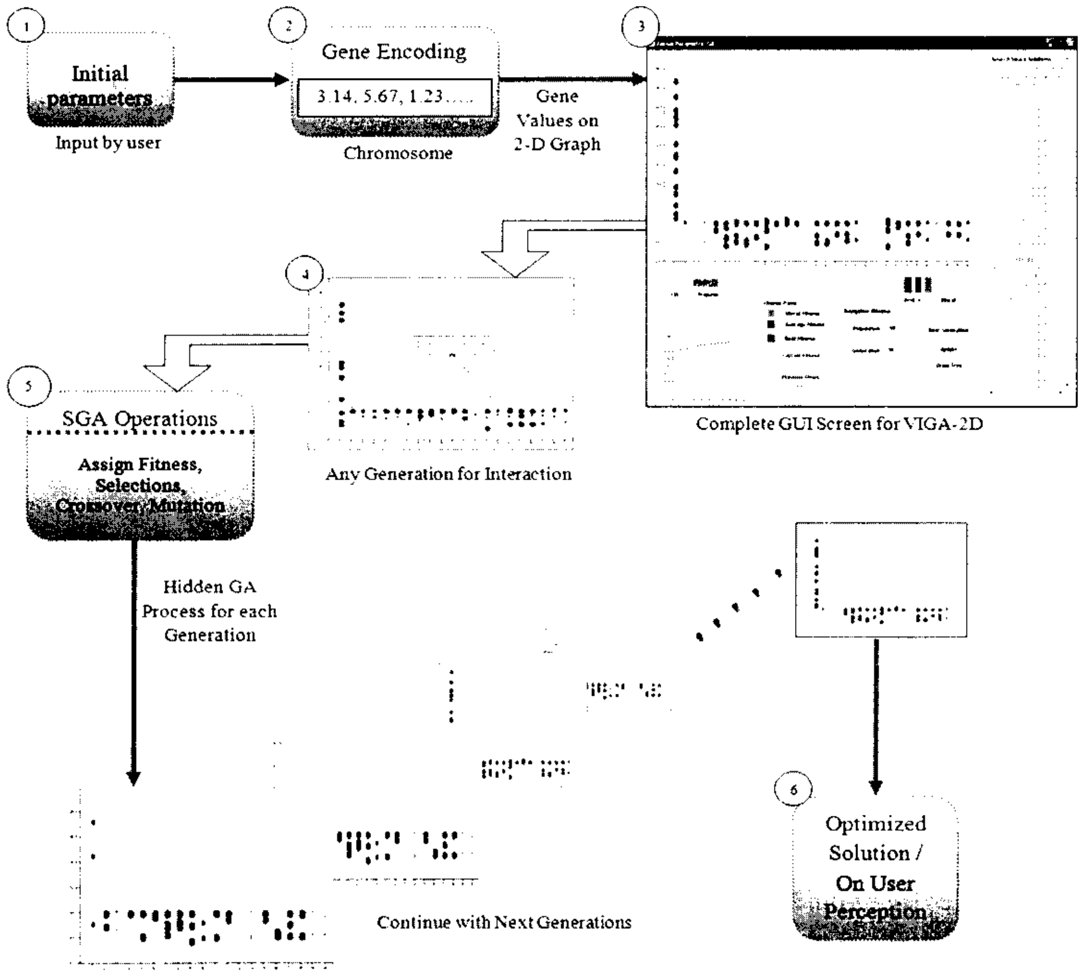

1
Initial parameters
Input by user
2
Gene Encoding
3.14, 5.67, 1.23.....
Chromosome
Gene Values on 2-D Graph
3
Complete GUI Screen for VIGA-2D
4
Any Generation for Interaction
5
SGA Operations
Assign Fitness, Selections, Crossover/Mutation
Hidden GA Process for each Generation
Continue with Next Generations
6
Optimized Solution / On User Perception

Figure 3.1: An Overview of VIGA-2D from Input to Output

The complete procedure of the proposed approach has been outlined in Figure 3.1. The first step is the encoding of a chromosome, which is based on the nature of problem (Test functions or the parametric values for the Parametric L-System). For test functions, the chromosome depends upon the length of chromosome string and its maximum and minimum value range. However, for evolving parameters for the Parametric L-System, it is randomly generated based on the user input string and initial parameters. After defining the problem to VIGA-2D, every generation is displayed on 2-D Graph. The user may go to next generation without interaction and he can do interaction in any generation. All operators i.e. selection, mutation and crossover works same as for SGA in the proposed approach. The user may select different selection method (Roulette Wheel Selection or Tournament Selection), crossover (1-point crossover or 2-point crossover), mutation rate and initial population size. However, with every successful interaction, there will be an increment in population size. With the multiple intervention of the user in search space, VIGA-2D converges to an optimal solution or a solution according to the human perception. For test functions the termination is based on generation number. Whereas, for deriving a rule for modelling the branching structures the termination is based on user perception. The next section describes in detail the visualization technique used to design a 2-D graph for a multidimensional search space for the proposed approach.

3.5 Multidimensional Data Projected on 2-D Graph

The distribution of multidimensional data on a 2-D graph makes it easy to grasp the data at different locations easily. In past, the attention of most of the researchers was based on the visualization of individuals of GA in 2 or 3-Dimensional [18, 19, 54]. Despite to these existing approaches, the proposed approach intends to distribute the gene values of each generation on to a 2-D graph. This allows the user to observe the searching behaviour of GA in lower dimension. On the hand, the interaction of user will be based on new gene values, as compared to existing works , where user's intentions was based on interventions with individuals [15,21] . In VIGA-2D, the horizontal view of the graph represents the gene locations and the vertical view represents the gene values of a chromosome. Hence, the data display on the graph depends on the range of gene values and chromosome length. This display of data involves the calculation for vertical and horizontal view of 2-D Graph and the visualization of retrieved gene values on 2-D graph. These calculations and techniques have been discussed in following subsections:

3.5.1 Vertical and Horizontal Ratio for 2-D Graph

For the representation of GA data on 2-D Graph, the first step is to calculate the ratio to set the data within the vertical and horizontal range of design graph. The vertical and horizontal ratios were calculated with the following equations:

Vertical Ratio: According to the division of the graph for vertical view, the data depends on the maximum range of gene values. The ratio for the vertical range is calculated according to the Equation (3.1).

$$v_{unit} = \frac{V_{length} - GAP}{m_{max}} \tag{3.1}$$

Where v_{unit} is the pixel value range along the vertical view, V_{length} is the total vertical length of the graph; *GAP* is a ratio to keep all the data on the graph and m_{max} is the maximum range of genes value.

Horizontal Ratio: For the horizontal view, the division of the graph depends on the chromosome length. The graph is equally divided into fixed length intervals. Hence, each interval represents each gene location.

3.5.2 Retrieving Gene Values from Each Individual

According to problem formulation each generation β consists of a set of individuals δ according to population size m and each chromosome is a set of gene values λ where l is the length of chromosome. Thus the first step involves retrieving the gene values λ from every location of chromosome in each generation as illustrated in Equation (3.2).

$$G_{value} = values_{m,l} \tag{3.2}$$

Where the G_{value} is a particular gene value, *values* is the range of the gene values, m= (0...population size) and l= (0....chromosome length).

Y-axis: After retrieving a particular gene value, the next step is to calculate the location of the y-axis, which is based on the calculation of v_{unit} as shown in Equation (3.3).

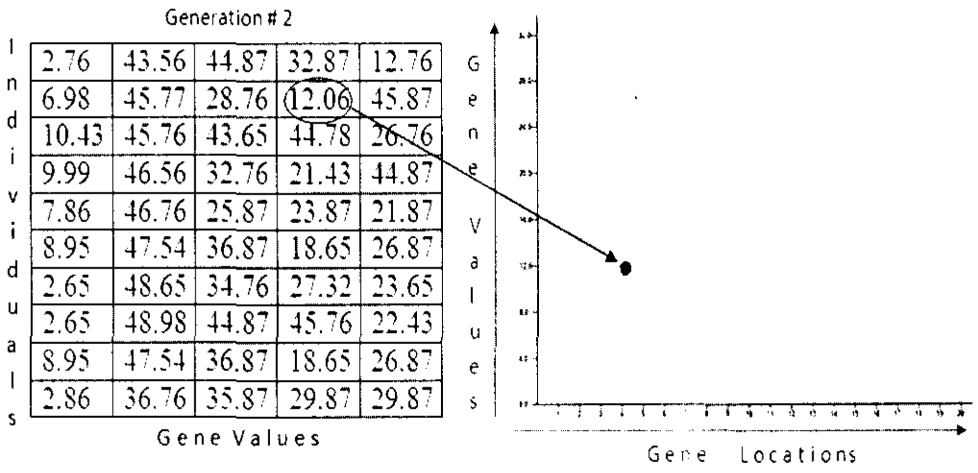
$$\text{y-axis} = V_{length} - (G_{value} - m_{min}) * v_{unit} \tag{3.3}$$

Where V_{length} is the total vertical length of the graph, the G_{value} is the gene value for display, m_{min} is the minimum range of gene values, and v_{unit} is the pixel ratio along the vertical view.

X-axis: The final step is to calculate the location of the x-axis, which is based on the calculation of the fixed interval as shown in Equation (3.4).

$$\text{x-axis} = \textit{x-axis} + \textit{interval} \tag{3.4}$$

Where the *interval* is a fixed value used to keep all data displayed according to the specific gene location. Hence, the calculated value is based on horizontal and vertical view along with x and y-axis.



2.76	43.56	44.87	32.87	12.76
6.98	45.77	28.76	12.06	45.87
10.43	45.76	43.65	44.78	26.76
9.99	46.56	32.76	21.43	44.87
7.86	46.76	25.87	23.87	21.87
8.95	47.54	36.87	18.65	26.87
2.65	48.65	34.76	27.32	23.65
2.65	48.98	44.87	45.76	22.43
8.95	47.54	36.87	18.65	26.87
2.86	36.76	35.87	29.87	29.87

Figure 3.2: A 2-D Graph for Visualization of Gene Values for Each Generation

3.6 Visualization on 2-D Graph

After setting the vertical and horizontal view of 2-D graph and retrieving gene values from each individual of a generation, the next step is involved with the visualization of data onto the graph. For visualization of a clear picture of the behaviour of searching the solutions in 2-D space, the fitness of gene values is displayed in blue level colours having a different colour depth with different sizes.

For example, the gene values with the best fitness are displayed with a dark colour (blue) having a larger size as compared to the gene values with worse fitness values which are displayed in a light colour (gray) with a small size. The user selects points (new gene values) according to the distribution of gene values with higher fitness for a better proposed individual. The geometrical shape "ellipse" is used to display gene values onto the graph. The significance of using different blue colour depths and size for different gene values is to make VIGA-2D self-explanatory to users. Hence, the user may observe the change in colour depth and size to take the decision for doing an interaction with current generation. On the other hand, this change in colour and size will also elaborate for understanding the gene distribution from generation to generation.

For the making a difference between the proposed gene values and the existing gene values, the proposed gene values are displayed in the red colour. In this way, calculated y and x-axis is responsible to displaying a single value as shown in Figure 3.2. This visualization of gene values are based on different blue colour and size based on their fitness in search space.

3.7 User's Interventions in Proposed Approach

User intervention is the main part of the proposed approach. In most of the existing IGA applications user's intentions are required in each generation for the selection of parents and evaluation of solution for assigning fitness, which creates a tiresome environment for the user. To overcome this tiresome problem of user, several techniques were introduced in the past i.e. the approximation of fitness values using a neural network [44, 45] or to assign a discrete fitness value for the evaluation of individuals [44]. However, these existing techniques worked with the fixed range of parameters and fitness values. Furthermore, in these existing applications the user selects the parents for the next generation.

In this proposed research work, the user may evolve many generations without interactions. The user is not involved with the selection of existing individuals. On the other hand, the designed interface for representing the gene values on 2-D graph also helps the user to decide the generation for interaction. For example, two different colours, blue and green, are used to demonstrate the current fitness. If the green colour is highlighted it means that the current generation has less average fitness as compared to the previous one, and the user may interact to improve the overall fitness.

If the highlighted colour is blue, it means that the GA searching is towards an optimal solution. Since an approach to interact with the GA searching process after several generations is adopted, based on the architecture of our problem, the searching process of GA is divided into two phases:

a) In the first phase, the user does not interact with the current generation. The fitness is calculated by the objective function, crossover, and mutation is performed and the next generation is created with a stable population size. Hence, this phase works in a traditional way for evolving the next generation with the visualization technique.

b) In the second phase, the user interacts with the current generation as shown in Figure 3.3. The interaction of the user is saved as a new individual (N_{child}). The N_{child} is passes to the objective function to calculate its fitness. The fitness of the proposed individual is compared with the average fitness ($F_{average}$) of the current generation. If the fitness of the proposed individual is higher than an average fitness of current generation then it becomes the part of search space in next generation, otherwise it is discarded by GA. Crossover and mutation is applied on the new population in the traditional way and the next generation is created with the plus-one strategy population size. This is illustrated in Algorithm 1.

```
Let $E_{pop}$ be the existing population with ($P_{size}$) size of population.
If the user interacts with current generation then
1. Save the new individual $N_{child}$.
2. Calculate the fitness ($F_{child}$) of $N_{child}$ using the objective function
        If ( $F_{child} > F_{average}$)
        Increase size of population: $P_{size}' = P_{size} + 1$.
        $E_{pop}' = E_{pop} + N_{child}$
         Else Discard the $N_{child}$.
3. Next generation.
```

Algorithm 1: Making the Proposed Individual as a Part of the Search Process

3.7.1 Retrieving Interacted Values as a Gene Values

Since the user interacts directly on the graph, this interaction is based on the pixel values. This pixel values work as new proposed gene values and act in the search space as a new individual. Each interaction is limited to the maximum and minimum range for a particular genes value. These pixel values are converted into data values (gene values) according to the calculation of x and y axis as given below:

Retrieving values from y-axis: The y-axis is involved in retrieving data according to the upper and lower bounds of each genes value. Equation (3.5) is used to convert pixel values into data values.

$$G_{value} = \frac{v_{length} - y_{pixel}}{v_{unit} + m_{min}} \tag{3.5}$$

Where the G_{value} is the gene value for display, V_{length} is the total vertical length of the graph, y_{pixel} is the y-axis position on the graph, v_{unit} is the pixel ratio along the vertical view, and m_{min} is the minimum range of gene values.

Retrieving values from x-axis: The x-axis is involved to specify the gene location for each gene value. The length of the chromosome may vary according to the input of the user. However, the maximum chromosome length was 30 for all experiments. As discussed above, the x-axis is divided into equal intervals according to the maximum length of the chromosome. These intervals are used to convert pixel values into data values for the x-axis.

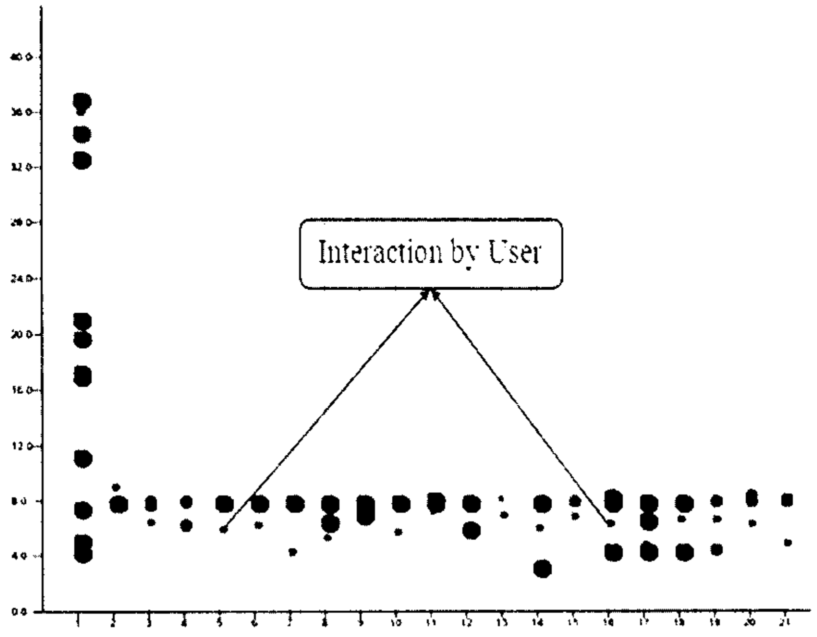


Figure 3.3: User Interaction on the Current Generation.

3.7.2 Dynamic Population Size for VIGA-2D

In the proposed research, each human interaction proposes a new individual instead of selecting any existing individuals for next generation; the problem was to make this new individual as a part of the search space in the next generation. A dynamic population size is proposed to overcome this problem. In this way the population size increases by one with the interaction of the user. In contrast to this, in most of the previous works, the issue was to control the size of the population with various approaches [47, 48]. In these existing works, variation (increment or decrement) of the population size depends on some other factors, i.e. fitness rating and threshold value. Variation in population size was also noticed in a IGA technique to model the fashion design on the screen , in which the population size increase or decrease with the interaction of the users [38]. .

In the proposed work, the main purpose of using the dynamic size of the population is not to lose the existing individuals but to enhance the search space interactively. Whenever user interacts with the current generation, the fitness of the proposed individual is calculated using the objective function. If the proposed individual is not optimal as compare to the existing individuals of the current generation, then the proposed individual discard by GA. The interactive tool helps the user to monitor the feasibility of GA to adopt the new individual in its searching process. For example, the increment in population size depends on the number of successful interactions that can be monitored by the user onto the interactive tool window. In this way, the user can observe the successful interaction with the increment in population size. In other words, if the user's interaction produces a worse individual, then the population size remains stable.

3.7.3 Visualization of Convergence Graph

Visualization of the fitness convergence graph works in a traditional way. A 2-D graph is implemented for monitoring the convergence of fitness in each generation. The graph is based on the visualization of the best, average and worst fitness value for each generation as shown in Figure 3.4. Importantly, this graph also helps to invites user for interaction, particularly, when the user observes no change in fitness from the last few generations, the user may interact with the GA process and propose a new individual to converge GA in a better way. This convergence graph also help to make the decision for terminating the GA process after achieving the desired convergence rate. In this graph the 'horizontal view' is used to display the generation number and the 'vertical view' is used to show the fitness value. Visualization of this graph is based on the number of generations being evolved and the difference in fitness value for each generation. For visualization of convergence graph, the vertical and horizontal ratios are calculated by the following equations:

Vertical Ratio: The vertical view of the graph depends on the maximum range of the fitness value. The ratio for the vertical range is calculated according to the Equation (3.6).

$$v_{unit} = \frac{V_{length} - GAP}{m_{max}} \tag{3.6}$$

Where v_{unit} is the pixel value range along the vertical view, V_{length} is the total vertical length of the graph, *GAP* is a value used to keep all data displayed inside the graph, and m_{max} is the maximum range of the fitness value.

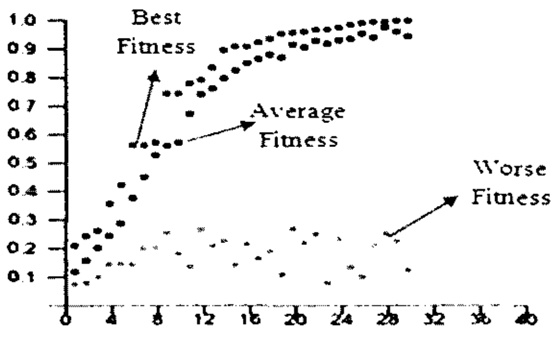


Figure 3.4: Fitness/Generation Graph: the Y-axis Shows the Fitness Values and X-axis shows the Generations.

Horizontal Ratio: The horizontal view of the graph shows the generation number. The ratio for the horizontal view is calculated according to the Equation (3.7).

$$h_{unit} = \frac{h_{length} - GAP}{h_{interval}} \tag{3.7}$$

Where h_{unit} is the pixel value along the horizontal view, h_{length} is the total horizontal length of the graph, *GAP* is a ratio to keep all data displayed inside the graph, and $h_{interval}$ is the intervals for displaying data in the x-axis according to the generation number.

X-axis: The maximum and minimum values of the x-axis are change according to the changes in the generation numbers. The x-axis is calculated according to the following Equation (3.8):

$x\text{-}axis = GAP + i * h_{unit}$ (3.8)

Where *GAP* is a value used to keep all data displayed inside the graph, *i = (0.. k)* is the generation number, and h_{unit} is the pixel ratio along the horizontal view.

Y-axis: For the y-axis, the higher and lower boundaries for fitness values are given by default. The graph is updated with every generation. Equation (3.9) shows the calculation for the y-axis.

y-axis = $V_{length} - (fitness_i - m_{min}) * v_{unit}$ (3.9)

Where V_{length} is the total vertical length of the graph, *fitness* is the fitness value to display, i= (0.. k) is the generation number , m_{min} is the minimum fitness value and v_{unit} is the pixel ratio along the vertical view.

The visualization of convergence graph shows the best, average and worse fitness for the each generation. Three different colours i.e. red, blue and green are used to display these fitness values respectively. The geometrical shape "ellipse" is used to display different fitness values. The size of the ellipse is the same for all fitness values.

3.8 GA Operators and Methods used in Proposed Approach

The VIGA-2D process starts in the traditional way as proposed by Goldberg [9]. Figure 3.5 shows that the initial population *(t)* is randomly generated. Then population *(t)* is evaluated using some criteria (fitness function) and this population enters into a loop and a new population is selected *(t+1)* from population *(t),* where crossover and mutation is performed, and after evaluation, a new population replaces the older one *(t).* And this loop continues until it meets the stopping criteria. The total number of individuals in a population shows the size of the population. However, during the searching process, human interaction may occur in any generation. As a result of interaction and proposing of a new individual, there will be an increase in population size. This interaction process is already discussed in detail in section 3.7. The following subsections will discuss the GA selection methods and different crossover and mutation operators applied for the GA process in this thesis.

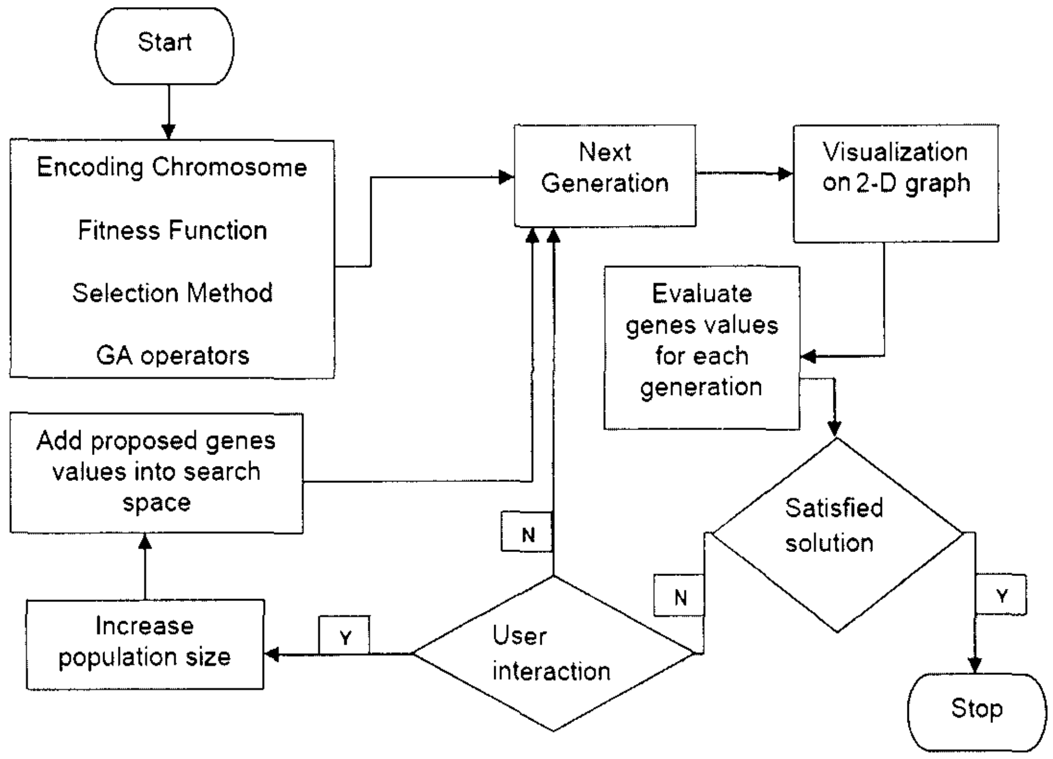


Figure 3.5: Pipeline for the Proposed Approach.

3.8.1 Selection

Selection is a process for choosing the best individuals to work as parents for the next generation. This selection is based on the fitness (selection pressure) of individuals. The best individuals are favoured according to a degree of selection pressure [105]. The higher the selection pressure, higher the chance for selection of the best individuals. The convergence rate is also determined by selection pressure. Thus the convergence rate becomes higher with a high selection pressure and GA finds the optimal solution faster [106]. As a result of selection, the chosen individuals are added to a mating pool in which the reproduction is occurs. In GA, different selection methods are used to provide selection pressure with different characteristics. Among them, the most common selection methods are used in the proposed approach; they are Roulette wheel selection and Tournament selection.

In the Roulette wheel selection method, the first step is to calculate the fitness ($F(c_m)$) for each chromosome (c_m).

The second step is to calculate the total fitness of the population as shown in Equation (3.10).

$$T_{fitness} = \sum_{m=1}^{pop\_size} (F(c_m)) \tag{3.10}$$

Where $T_{fitness}$ is the total fitness and $m = 1, 2 \ldots\ldots\ldots, pop\_size$ and $(fitness\ (c_m))$ is the fitness of the chromosome in the population. An empty mating pool is created and filled with the selected individuals based on their fitness value. The fitter individuals have a higher chance to be selected and added into the mating pool. A new population is generated based on the fitness value with respect to probability of distribution.

The third step is to calculate the selection probability p_m for each chromosome c_m using equation (3.11).

$$p_m = \frac{F(c_m)}{T_{fitness}} \tag{3.11}$$

The fourth step is to calculate the cumulative probability q_m for each chromosome c_m using Equation (3.12).

$$q_m = \sum_{j=1}^{m} p_j \tag{3.12}$$

Adding the fitter individuals to the mating pool are continued until the population is full, for this process, a random number r is generated.

If $r \leq q_1$, then the first chromosome c_1 will be selected

Else

Select k_{th} chromosome c_m such that $q_{m-1} < r < q_m$.

After selection of individuals for the new population, the recombination and mutation occur.

Tournament selection is based on a tournament between members (i.e. an individual of the population) to provide a selection pressure [105]. This selection method is computationally more efficient. These individuals are selected from the population randomly. The tournament selection pressure is increased by an increase in tournament size (T_s) or vice versa. The winner is the individual having the higher fitness among T_s and it is added to the mating pool. It involves the following steps:

1. With the probability *p*, selects the best individual from the tournament / pool.
2. Select the second best individual

   p*(1-p)

3. Select the third best individual

   p*((1-p)^2)

The tournament selection remains continue until the mating pool is full. A new population is created by selecting two random individuals as parents from the mating pool. After performing crossover on these parents, a child is created. Mutation is applied on the created child and a new population is created for the next generation.

3.8.2 Crossover

Recombination or crossover is an act of exchanging information or genetic material between parents to produce new children. This exchange is based on selection of better genes to produce good solutions referred to as "building blocks for the next generation". For reproducing well adapted individuals with new genetic material, two types of crossover operators are performed in the proposed approach, i.e. one point crossover and two point crossover [108].

3.8.3 Mutation

Mutation also plays an important role during the GA process to bring new features into the next generation [109]. It occurs by giving some probability value during evolution. Mutation is used to alter current alleles of genes with different alleles. The position for mutation is selected randomly. A mutation rate m_t is defined. For example if m_t is 0.01, then only 1% of the genes in the population will be mutated. The mutation is performed on the genes which are less than the mutation rate m_t . Every gene has an equal chance to be mutated. After completion of the mutation, the new child (individual) is added to the population.

In the proposed work, the uniform mutation [110] is used. Uniform mutation works with the alteration of genes by the selection of a uniform random value within the upper and lower bound of a particular gene. The next population is generated after one completes process of selection, crossover and mutation. Thus the one generation of GA depends on the selection, recombination, mutation and evaluation. After mutation, a new child is added to the population $\beta(t+1)$.

3.8.4 Termination Criteria

The termination condition is an important part of GA, which helps the algorithm to decide on going to the next generation or stopping the evolution process. This termination criterion is checked after each generation. There are many ways to terminate the GA search process depending on the nature of the problem, i.e. number of generations, computational time, and threshold value or according to the user's perception [2]. In IGA, it is very difficult to determine termination criteria theoretically. The applications of IGA are user dependent, so it depends on the user to decide the termination of the current evolution. The decision of the user for termination may depend on the convergence of fitness to optimized solution or it may depend on the user's requirement or the number of generations evolved. In the proposed work, two termination criteria are used: (1) Termination of the process depends on the user. (2) Number of generations.

3.9 Application: Modelling the Growth Process of Branching Structures using Parametric L-System

Parametric L-System has been proved to be a successful rewriting mechanism to demonstrate the growth process of plants and trees [82]. The application of the proposed approach is to optimize the parameters for deriving the rules for the Parametric L-System while symbols for the structure are given by the user. The best solution obtained from VIGA-2D works as associated parameters, used for deriving rules for the Parametric L-System.

The deriving of rules for Parametric L-System using GA is a further extension to the work done in [86], in which a sketch interface is developed to retrieve an L-System string from user's sketch for a tree. The stroke input is translated into L-System symbols and parameters, which indicates the height, main axis and a number of iterations from which the tree should be grown. The rules of the L-System are developed with the help of user's input that is used to model the further growth of the tree according to the number of iterations. For deriving the parameters closest to the input of the user; GA is used to optimize the parameters. The rules derived from the optimized solution are then used to generate branching structures using the turtle interpretation onto 3-D space.

In the proposed work, the process initiated with the L-System string given by the user. Initial parameters for the branching structure, i.e. the branching angle, length and width of trunk are given by the user. Whereas the parameters used for scaling and deriving the rules are randomly generated by the system. Initial parameters given by the user and randomly generated parameters works as a target solution in VIGA-2D. All these parameters work as real values for chromosome encoding in VIGA-2D. User can do multiple interactions for proposing a new individual and user may see the output (generated branching structure) in any generation. The overall process works in the same manner as discussed in section 3.4. In order to visualize the growth process of generated structure, the optimized solution from the last generation acts as deriving rules for Parametric L-System. The steps involved in the process are shown in Algorithm 2.

```
1: Initialize the process with user input (L-string) and initial parameters
2: Generate random chromosome from initial parameters (Oparam) to initiate the VIGA-2D process.
3: Calculate the fitness of each GA generated solution (Gparam) as compare to Oparam with fitness function:
fitness = $\sqrt{\sum_{i=1}^{l}(Gparam_i - Oparam_i)^2}$
Repeat
4: Until terminate according to user perception
5: Derive L-System rule from best solution obtained from current generation.
6: Model the next growth step according to the L-System rule
```

Algorithm 2: Steps taken for Deriving Rules with L-string and VIGA-2D.

In order to generate branching structures using VIGA-2D, the graphical interface is divided into 3 stages. The first stage is involved with the user input for generating L-System string. The second stage is involved with the VIGA-2D search process for optimizing the parameters and the third stage is responsible for deriving rule and the visualization of the branching structure in 3-D space as shown in Figure 3.6. The termination of the VIGA-2D searching process depends on the user. During visualization, the fitness graph shows the overall performance of the process as discussed in subsection 3.7.3. Besides displaying branching structures repeatedly, the user may estimate the performance of finding a good solution through this graph. The following subsection will discuss these stages architecture and the functionality of each stage in further detail.

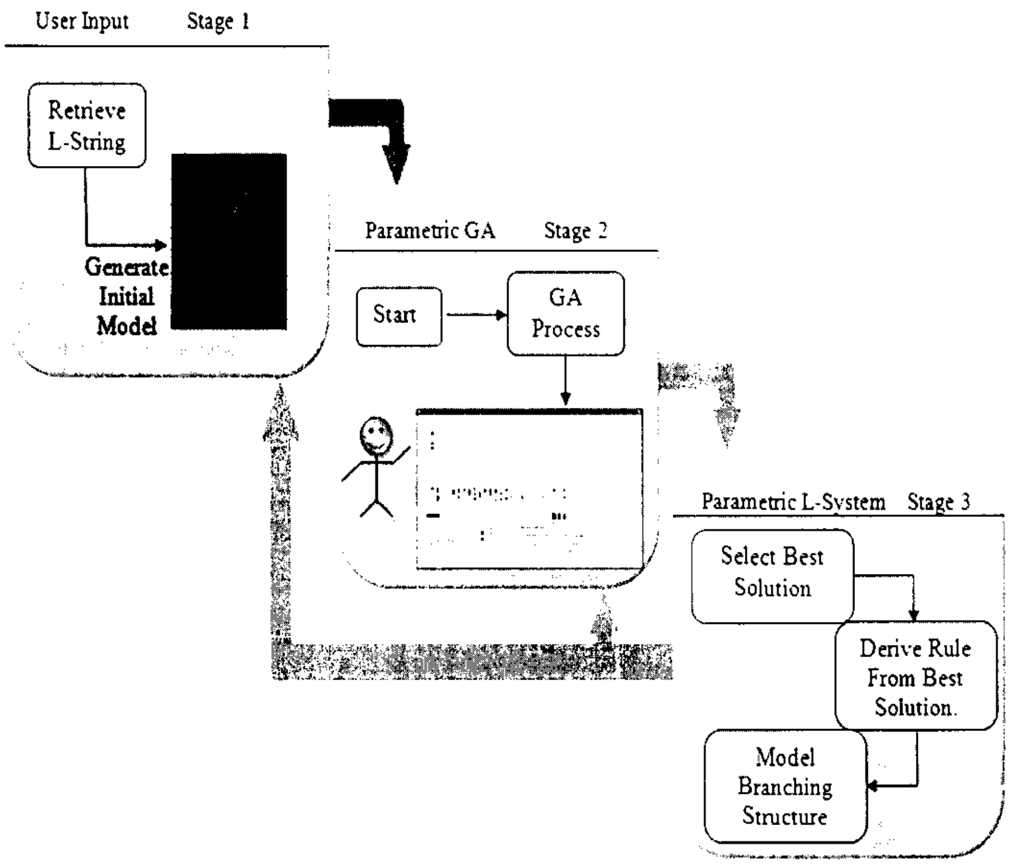


Figure 3.6: An Overview of the Designed Interface for Deriving the Rule for Parametric L-System.

3.9.1 User Input (L-String).

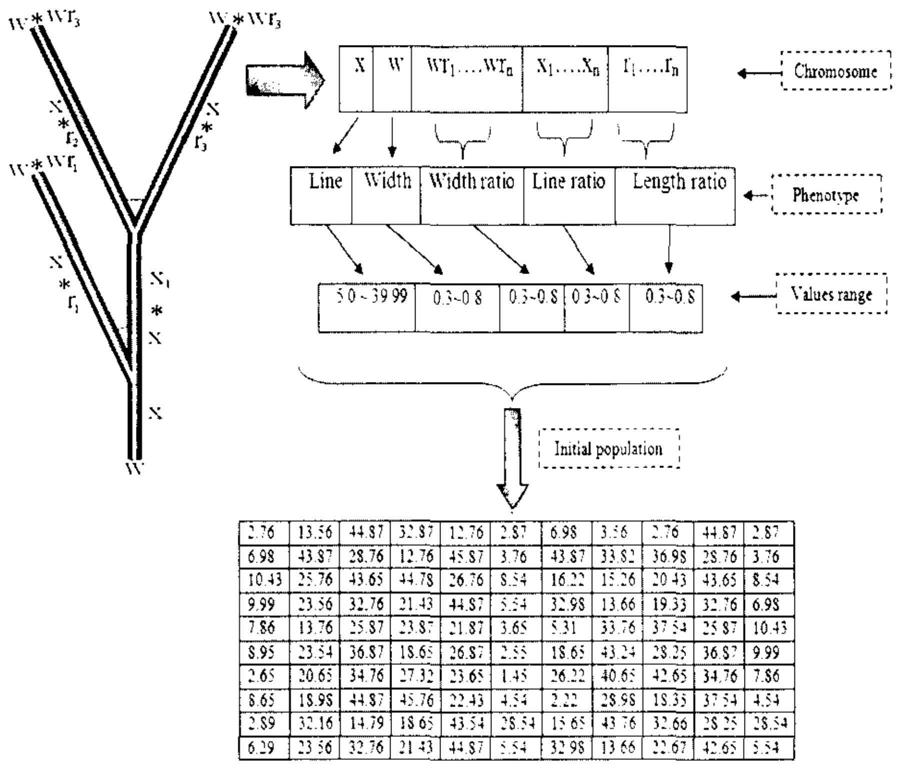
The L-System language consists of terminal and non terminal symbols. Non terminal symbols known as repeated symbols are used to generate more strings of symbols. Terminal symbols are used for scaling, rotation and movement, and they remain same during all iteration levels. The bracketed L-System is applied to generate tree like structures. In the proposed work, the Deterministic Parametric L-System is used to create the rule according to the following form:

Predecessor → Successor

Where the predecessor will be recursively expanded and replaced with the relevant successor in each iteration. The graphical interpretation is based on the turtle interpretation [80]. In the proposed work, the turtle interpretation is based on as discussed in [111, 112). The three dimension coordinating system is adopted with 3 vectors (H, L, and U) used for directions, in which, H represents the head or front vector, L represents the left vector and U represents the up vector. The default angle is H at 90 degree (see Appendix B for detail).

In the proposed approach, the problem is to optimize the parameters of Parametric L-System for deriving the rule. For this reason, it covers the translated symbols from the user input and their initial parameters. These symbols are based on the user input in all 3 directions. The following example illustrates the mechanism of generating the random chromosome to initiate the VIGA-2D searching process.

Let the input string given by the user is =!F*[+]F[-]*. The total numbers of symbols used in this example is 7. After modelling the initial structure with the user input, the L-String gets the following form: *!F[+!F[+]F[-]]F[-!F[+]F[-]]*. For the initial structure, width of the trunk, height of the main branch and angles of the sub branches are also input by the user. The next step is to create the random parametric values which work as a chromosome for VIGA-2D. The length of the chromosome is based on the total number of symbols in the L-String including the width and length for each sub branch, represented by the symbol A.



2.76	13.56	44.87	32.87	12.76	2.87	6.98	3.56	2.76	44.87	2.87
6.98	43.87	28.76	12.76	45.87	3.76	43.87	33.82	36.98	28.76	3.76
10.43	25.76	43.65	44.78	26.76	8.54	16.22	15.26	20.43	43.65	8.54
9.99	23.56	32.76	21.43	44.87	5.54	32.98	13.66	19.33	32.76	6.98
7.86	13.76	25.87	23.87	21.87	3.65	5.31	33.76	37.54	25.87	10.43
8.95	23.54	36.87	18.65	26.87	2.55	18.65	43.24	28.25	36.87	9.99
2.65	20.65	34.76	27.32	23.65	1.45	26.22	40.65	42.65	34.76	7.86
8.65	18.98	44.87	45.76	22.43	4.54	2.22	28.98	18.35	37.54	4.54
2.89	32.16	14.79	18.65	43.54	28.54	15.65	43.76	32.66	28.25	28.54
6.29	23.56	32.76	21.43	44.87	5.54	32.98	13.66	22.67	42.65	5.54

Figure 3.7: Representation of the Parameters for Chromosome Encoding

3.9.2 Evolving Parameters using VIGA-2D

The deriving of parametric rules for Parametric L-System is based on the different parameters. The genetic encoding of these parameters is real values, which works as a chromosome for VIGA-2D. Figure 3.7 shows the construction of the chromosome according to the parameters retrieved from the branching structures. Each of these parameters represents different gene values and has a particular range. A random generator is used to assign random values according to upper and lower bounds for different gene values.

The chromosome length is a dynamic length and depends on the number of symbols given in the input L-System string (L-String). The searching mechanism is indirect with a non-linear equation. As real values are evolved, the GA process evolves until it eventually converges to the nearest optimized solution or the best solution. Based on the formal parameters; the fitness function is defined as follows:

Evaluation Function: The optimal or nearest real values are measured by summing the squares of the differences between the indexing of the original values at each position and the values generated by GA. This is represented by Equation (3.13).

$$f = \sqrt{\sum_{i=1}^{l}(Gparam_i - Oparam_i)^2} \tag{3.13}$$

Thus *f* will be the fitness value of the current evolving individual, *Gparam* is the GA generated solution and *Oparam* is the target solution and i=(0.. *l*) is the length of the chromosome. This fitness function evaluates the fitness measured for each individual indicating its suitability to be selected.

3.9.3 Visual Representation of Parametric Values

The visualization is involved to represent gene values for every generation on the screen. Every gene value has a particular range in constructed chromosome. Equation (3.2) is used to display the gene values on the screen. Figure 3.8 shows the 2-D graph for the parameters. The gene locations of the genes value on the graph are displayed in the same the sequence as in the chromosome. Each gene has a particular meaning in phenotype form and a particular range of values. For example, the first index is allocated to the line parameter which is the 'F' symbol having the range from 5.0 – 39.99, the second index is allocated to the width parameter which is the '!' symbol having the range from 3.0 – 8.0 and all remaining gene values range are also in the range between 0.3-0.8. To display all gene values in equal scale, all genes value except the first one are scaled up to 3.0-8.0 for visual purposes.

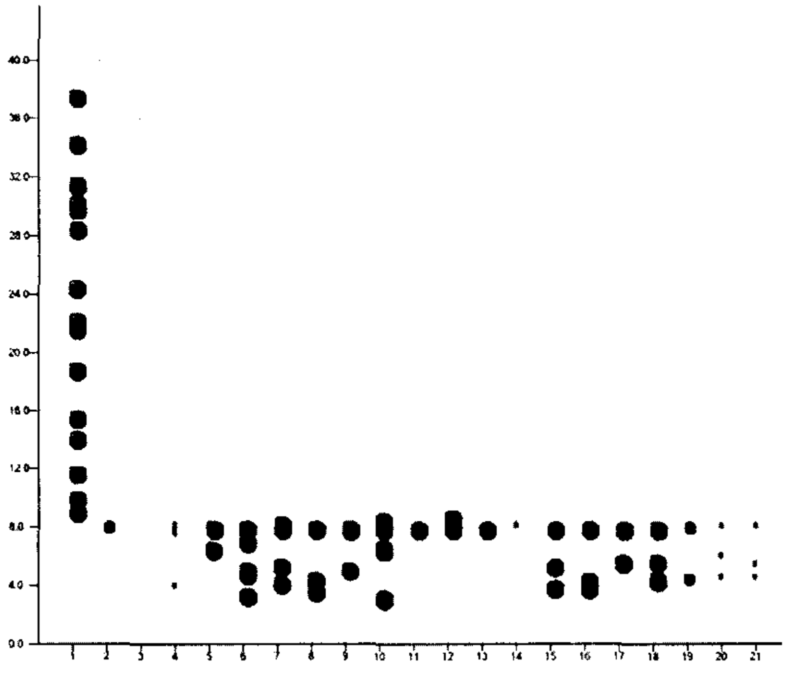


Figure 3.8: Visualisation of Gene Values on a 2-D Graph for Parameters

3.9.4 Deriving Rule for the Parametric L-System

Termination of VIGA-2D is based on the user perception. After termination, the best solution obtained from the last generation of VIGA-2D works as parameters for L-System string to derive the rule for the Parametric L-System. For deriving rule, the parameters are combined with the symbols, based on the gene location and its phenotype meanings, i.e. according to the symbols. The rules of the L-System are based on two parts which are the axiom and the rules. The axiom is known as the starting point of generative grammar. In the presented application, the axiom is constant for rule. It is implemented with one symbol "A" with corresponding parameters as follows:

$$\omega = axiom\ (r,g,b)A(x,w)$$

Where,(r,g,b) are the colour values for Red, Green and Blue, "A" is used for a recursive purpose, x shows parametric value for the line symbols "F", and w shows the parametric value for width symbols "w".

For the L-System, rules can vary in numbers, depending on the complexity of the generated structure. We developed one rule from the output of VIGA-2D. This rule is applied to the non-terminal symbol of the axiom. The construction of this rule is as follows:

$$A(x,w) \text{ -> } !(w)\ F(x)\ [+(\theta_1)A(r_1,wr_1)\ ]\ F(x_1)\ [\ +(\theta_2)\ A(r_2,wr_2)]$$

$$\ldots\ldots\ F(x_n)\ [\ \&(\theta_n)\ \ A(r_n,wr_n)]$$

In the above rule, the parameters, i.e. $r_1, wr_1, r_2, wr_{2...} r_n, wr_n$ are randomly generated for the chromosome; whereas, the value for the parameters, i.e. x and w are given by the user. The value for θ is constant for all angles. Following are the assumptions and limitations for deriving the rule for generating the branching structures.

1. For assigning the parameter to the symbol, *( )* are used, i.e. "! *(2.53)"* shows that the w*idth* of the branches is *2.53*.

2. To separate the predecessor from successor "-> "is used.

3. The predecessor may also receive a variable using ( ), i.e. A *(x,w)*. This variable is used in the successor as parameter.

4. The number of iteration shows the complexity of the generated structure.

5. In the production rule, the *A* behaves recursively for *n* number of iterations, where the value of *n* is the number of iterations.

For example, if the L-String given by the user is *!F[+]F[-]*, *the* angle for each branch 'θ ' is 22 and the initial parameters are *37, 4 and 28*. Then, after the initial structure, the L-String becomes: *!F[+!F[+]F[-]]F[-!F[+]F[-]]*. For VIGA-2D and the length of the random chromosome will be 21, based on the total number of symbols in the L-String. Hence, the randomly generated chromosome for the above L-String will be : *37,4,28,20,5,31,25,9,30,11,20,19,8,24,33,29,16,12,24,7,9*.

After evolving the VIGA-2D for n-number of generations, the following is the derived rule with integration of the L-String and the best solution obtained from VIGA-2D:

```
w          (0.24,0.6,0)A(19,8)
# A(x,w)-> !(w)F(x)[+(22)$A(x*0.81,w*0.81)!(w*0.81)
F(x*0.81)[+(22)$A(x*0.81, w*0.51)]F(x*0.35)[-(22)
$A(x*0.61,w*0.21)]]F(x*0.81)[-(22)$A(x*0.53,w*0.
.65)!(w*0.31)F(x*0.69)[+(22)$A(x*0.72,w*0.88)]
F(x*0.76)[-(22)$A(x*0.51,w*0.45)]]
```

In this way, every generation in which the user wants to generate a tree is used to derive a rule from its best solution. This searching process remains continue until user terminates it.

3.10 Test Functions with VIGA-2D

In this thesis two test functions were examine in order to test the performance of the proposed approach and to evaluate the efficiency as compare to SGA . These two test functions are De Jong's [113], and Rosenbrock function [114]. Table 3.1 shows the names of the functions with its objective function and the limits of the upper and lower bounds. Chromosome encoding for these test functions is real value encoding. GA operators and selection methods work in the same way as discussed in Section 3.4.

Table 3.1: List of Test Functions and their Limitations Evolved with the Proposed Approach

Function Name	**Fitness Function**	**Limits**
De Jong's F1	$$f(x) = \sum_{i=1}^{n} x_i^2$$ And the global minimum is: f(x)=0, x(i)=0, i=1:n.	[-5.12,5.12]
Rosenbrock	$F_2 = \sum_i^{n-1} 100(x_2 - x_1^2)^2 + (1 - x_1)^2$ And the global minimum is: $x_i = 1$, f(x) = 0 , i=1:n.	[-5.12,5.12]

The visual representation of gene values for test functions works in the same way as for parameters of Parametric L-System (see Section 3.5.2). The equation for displaying these gene values is calculated according to Equation (3.2), where the *value* is according to the minimum and maximum range given by the user.

3.11 Summary of Chapter

This chapter has proposed a new approach to visualize the multidimensional data of GA in a 2-D space. The implemented 2-D graph represents the visualization of gene values and provides a way to interact the GA process by the user. In proposed approach, the intervention of the user is not involved with the selection and assigning of fitness to the existing individuals. The purpose of this interaction is to propose a new individual, based on different gene values selected by the user during the evolution process. This proposed individual is within the upper and lower bounds of the gene values and works as a new born child in the next generations.

For making the user's interventions effective, different visualization techniques, i.e. the usage of different colours and size for displaying the gene values, displaying the average fitness value of the current generation on the screen and the visualization of convergence graph are used to help the user in understanding the searching process. An objective function is used to calculate fitness for the solutions in each generation. Therefore, user interventions are not forced in every generation.

This chapter also shows in detail the working mechanism of the proposed approach to derive the rule for the Parametric L-System. This rule is used to model the growth process of the branching structure according to the user input. For testing purpose, two benchmark functions are selected. The success of an approach depends on complete and accurate evaluation. The next chapter will discuss the experimental results based on evolving the selected benchmark functions and evolving parameters for deriving rule for the Parametric L-System.

CHAPTER 4

EVALUATION OF RESULTS

4.1 Chapter Overview

This chapter evaluates the proposed approach as discussed in Chapter 3. For effective and thorough evaluation of the proposed approach, a range of different experiments have been carried out. These experiments were based on different inputs, number of interactions, length of chromosomes, and a number of generations. In order to analyse the convergence of VIGA-2D, it was studied through different experiment to observe it convergence at an optimal or sub-optimal solution. The main purpose of proposed approach is to accelerate the performance of GA as it has been addressed in several existing techniques [15, 44, 26]. For this reason all the experimental results were compared to SGA in order to evaluate the ability of the proposed approach to converge to an optimized solution or to best solution according to the user perception.

In contrast to the previous approaches, the proposed research involves the visualization of multidimensional data on a 2-D graph based on gene values. In the proposed approach, user interaction is based on proposing a new individual to accelerate the convergence of GA towards optimal solution. The main idea of proposed approach is to find an optimal solution with several interactions in fewer generations having small population size. The fitness is calculated by the fitness function; hence, a human-machine combination allows this approach to be applied successfully to different problems. As human interaction is the main part of the proposed methodology, most of the discussion in this chapter will be on user interactions and their effects on the GA searching process.

To demonstrate the performance and efficiency of the proposed approach for an optimal solution, the evaluation results are presented in Section 4.4 with a discussion of applying the proposed approach for benchmark functions. For this purpose, two benchmark functions were chosen with different features. First function selected was DeJong function [113]; converge at global optima. Second function was Rosenbrocck function [114] having many local optima and difficult to converged. In this way the proposed approach has been evaluated with two different kinds of functions. However both of these functions are subjects of minimization. For analysing the performance of VIGA-2D, for evolving parameters for deriving rule for the Parametric L-System are presented in Section 4.5. Based on the nature of the proposed approach, the results are mainly divided into the following two 2 categories:

a) **Experiment 1** (Objective Analysis): The performance of VIGA-2D was evaluated with benchmark functions. For comparative evaluation, the same GA parameters and operators, i.e. length of chromosome, size of population, number of generations, selection method, crossover and mutation rate were used to evolve the SGA. Hence, difference of convergence rate of VIGA-2D was compared with SGA. The comparison between SGA and VIGA2D has been done with same number of generations.
b) **Experiment 2** (Subjective Analysis): The performance of the proposed approach was evaluated with the user. Five different users run VIGA-2D with different chromosome lengths, selection methods, operators and number of generations to evolve the parameters. These parameters were used to derive the rule for the Parametric L-System. This rule was then used to model the growth steps of branching structures according to the user input. The termination of searching process was based on the user perception.

After this, some experiments were conducted to run VIGA-2D with different selection methods and operators to evaluate the advantages of using the visualization technique for the searching process of GA. Following are some observations which will remain constant for all experiments.

- Analysis of user interactions
- Evaluate the convergence rate based on the best and average fitness of each generation.
- Explore the gene values according to given colour scheme.
- GA progress towards a better or optimized solution

4.2 Specification of the System and Dataset

The experiments were conducted on an Intel(R)Core(TM)2 Quad CPU Q6600 running at the CPU speed of 2.40 GHz with a 3.0 GB RAM. The operating system was a Microsoft Window XP Professional edition version 2002. For human perception, the interfaces and frames were best viewed on an LCD with resolution of 1024 X 768 pixels. In all experiments, JDK 1.6 was used to run the proposed program developed in JAVA.JAVA 2 and 3-D graphics version 1.5.1 is used to implement different graphical modules.

4.3 Benchmark Functions (Objective Analysis)

For evolving the benchmark functions, as described above, the selection method, operators, crossover and mutation rate remained the same for all experiments. The selected parameters for evolving VIGA-2D and SGA i.e. selection method, parameters, rate and operators for benchmark functions was based on several testing of proposed approach and SGA with different parameters. After analyzing and evaluating the performance of VIGA2-D and SGA with these different parameters, following operators and selection method has been set for all experiments for both algorithms.

- Roulette Wheel Selection method was used
- One point crossover was performed
- Mutation was performed with the mutation rate of 0.05.

- Termination condition was the number of generations
- Each experiment was performed 10 times
- Chromosome length used for the experiments was n = 10 and 20
- Two graphs were drawn for comparison between SGA and VIGA-2D
    1. Average Fitness (Total average of all averages of 10 runs each).
    2. Best Fitness (Total average of the best fitness of 10 runs each)

Each experiment was performed 10 times with same chromosome length. The population size was 10 for chromosome length n=10. Whereas for chromosome length n=20, population size was 20. Each experimental result was performed with 10 interactions for keeping consistency in all experiments. In a total 100 generations was evolved for every run for VIGA-2D, however, SGA was evolved to a number of generations to get the closest optimized solution as compared to VIGA-2D. For visualization aspects of VIGA-2D, the curve of convergence rate is directed to upward direction due to implementation aspects. However, both functions are the subjects of minimization. The discussion of all experiments will be based on average result calculated after 10 runs for both benchmark functions. The Appendix C shows the detail tables for interactions, accepted and discarded proposed individuals with both functions.

4.3.1 DeJong's Function

It is the first function of De Jong's [113], known as sphere model. This function is smooth, continues, convex, unimodal, symmetric and converge at global optima. For evaluating the performance of the proposed approach with De Jong's Function, experiments have been done with SGA and VIGA-2D. Table 4.1 shows all important parameters taken for DeJong's Function.

Table 4.1: De Jong's Function Specification

De Jong's Function	n= 10	n=20
Runs	10	10
Generations	100	100
Population size	10	20
Total Interactions	10	9
Average Accepted Interactions	8	9
Average Discarded Interactions	2	1
Average Population Size after Interactions	18	29

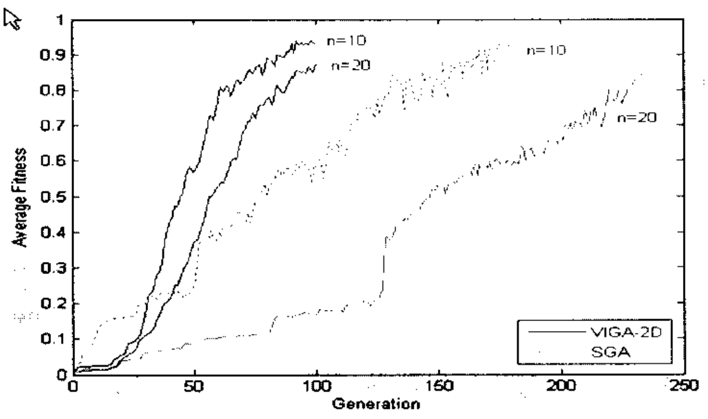


Figure 4.1:(a) Average Fitness / Generation Graph

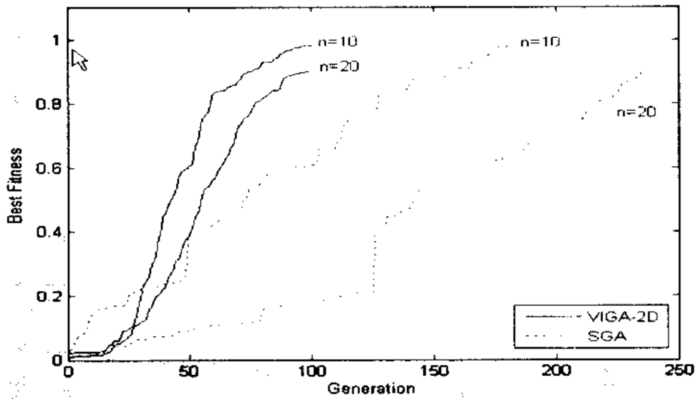


Figure 4.1: (b) Best Fitness / Generation Graph

Figure 4.1: Fitness / Generation Graph for De Jong's Function for n=10 and 20.

Figure 4.1 shows the fitness / generation graph with VIGA-2D and SGA. In this experimental result it was noticed that until the generation 20 the convergence of SGA and VIGA-2D was at the same rate. The first average interaction was done at generation 17 for all runs. The second interaction was done at generation 23 for all runs. After these two interactions there was a prominent difference noted between the fitness value of VIGA-2D and SGA. The last interaction was done at the average generation of 83 for all runs, with a total of 10 average interactions. The average population size was 18 at termination time. On average, 2 proposed individual was discarded by VIGA-2D.

For n = 20, the population size was 20 and the generations size was 100 for VIGA-2D and 236 for SGA. The convergence of VIGA-2D was higher as compared to SGA after the 1st interaction. The average first interaction was done at the generation 17 for all runs. In total, 10 interactions were done with the average population size of 29. On average, 1 proposed individual was discarded by VIGA-2D. The last interaction was done at the average generation of 90 for all runs.

Table 4.2: Comparison between VIGA-2D and SGA after 10 runs for De Jong's Function upto 100 Generations

	n=10		**n=20**	
	Best Fitness	Average Fitness	Best Fitness	Average Fitness
VIGA-2D	0.979798	0.92806	0.90073	0.872481
SGA	0.609368	0.58445	0.18287	0.169025
Difference	0.37043	0.34361	0.71786	0.689611

A prominent difference was noticed while evolving this function with and without VIGA-2D. For keeping consistency in Table 4.2, difference noted was based on generations 100 for both algorithms. For n =10, the difference between SGA and VIGA-2D was found to be *0.34361* for the average fitness. Whereas the difference of both algorithms at n=20 was *0.689611* for the average fitness. Table 4.2 shows that the proposed approach was successful to converge with the best fitness of *0.979798* for n =10 and *0.90073* for n = 20 after 100 generations. However for SGA approximately same difference was noted after 181 generations for n=10 and after 236 generations for n=20. Hence, these differences in the convergence rate and generations show that the interaction at different generations with the proposed approach helps the De Jong function to optimize much faster as compare to SGA with a small population size.

4.3.2 Rosenbrock's Function

The Rosenbrock function [114] is a non-convex classic optimization function. It is also known as banana function or the 2nd function of De Jong. This function is frequently used to analyse the performance of optimization problems. The global minimum is inside a long, narrow, parabolic shaped flat valley. To find the valley is trivial, however convergence to the global optimum is difficult. Table 4.3 shows all important parameters taken for Rosenbrock's function for the evaluation of the proposed approach as compared to SGA.

Table 4.3: Rosenbrock's Function Specifications

Rosenbrock's Function	**n = 10**	**n=20**
Runs	10	10
Generations	100	100
Population size	10	20
Total Interactions	10	10
Average Accepted Interactions	7	7
Average Discarded Interactions	3	3
Average Population Size after Interactions	17	27

Figure 4.2 shows the fitness/ generation graph for Rosenbrock function. For n=10, the first average interaction was done at the average generation number 17 for all runs. With the average of 7 successful interactions for VIGA-2D converging with a higher fitness rate as compared to SGA. In total, 10 interactions had been done. The population size was 10 and a total of 100 generations were evolved.

For n=20, the first average interaction was done at the average generation number 16 for all runs. With the average of 7 successful interactions, VIGA-2D converges with a higher fitness rate as compared to SGA. On average, 3 proposed individuals were discarded by GA. The population size was 20 and the total generations evolved were 100.

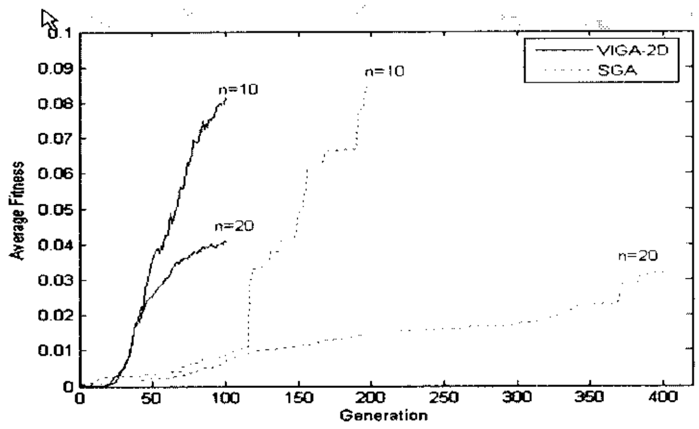


Figure 4.2: (a) Average Fitness / Generation Graph

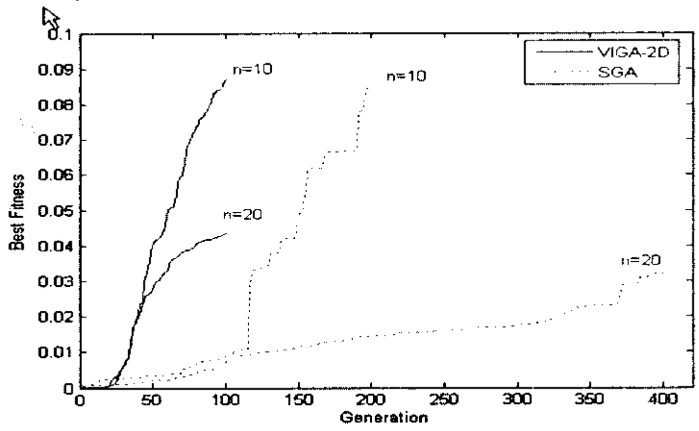


Figure 4.2: (b) Best Fitness/ Generation Graph

Figure 4.2: Fitness/Generation Graph for Rosenbrock's Function for n =10 and 20.

Table 4.4: Comparison between VIGA-2D and SGA after 10 runs for Rosenbrock's Function upto 100 Generations

	n=10		**n=20**	
	Best Fitness	Average Fitness	Best Fitness	Average Fitness
VIGA-2D	0.08694	0.081295	0.04345	0.04038
SGA	0.008627	0.008568	0.00728	0.005968
Difference	0.078313	0.072727	0.03617	0.034412

The graphs in Figure 4.2 (a) and (b) show a difference between fitness convergence of SGA and VIGA-2D. After 10 times of execution the difference for average fitness between VIGA-2D and SGA for n = 10 was 0.072727 and for n = 20 it was *0.034412*. Table 4.4 lists all the best and average fitness found with the two different variable lengths after 100 generations. It was noted that for SGA with n=10, Rosenbrock function converged to nearest fitness value to VIGA-2D after 200 generations and for n=20 it was after 400 generations. Due to multimodal feature of Rosenbrock function both of algorithm were not successor to a good fitter value. Overall the performance of VIGA-2D was satisfied as compared to SGA with different generations in presented experimental results.

4.4 Discussion with Benchmark Functions

In the presented experiments, the performance of VIGA-2D was compared with SGA along with two benchmark functions. The observation was based on the convergence rate of VIGA-2D towards an optimized solution with human interactions and without interaction for the convergence rate of SGA.

The results show that the interaction of humans brings a prominent difference between convergence rates from generation to generation. It was noticed that proposing new individuals in several generations brings a prominent change in the searching process. Additionally, in the proposed approach an increment in population size after each successful interaction also gives a wider search space, which helps VIGA-2D to converge actively as compare to SGA. However it was noted that for De Jong's Function due to its unimodal feature , there was not a large difference in convergence with n = 10 and 20. Overall, the performance of VIGA-2D was better than SGA for converging towards a fitter solution in different generations.

Test function Rosenbrock also showed a better performance with VIGA-2D for both variable lengths as compare to SGA. Although the convergence with VIGA-2D has not shown a good performance for this function, the overall performance was satisfactory as compared to SGA. For VIGA-2D, the performance of the rosebrock function was noted at the same convergence for n = 10 and 20. A difference was noted after 45 generations, although there were 5 average interactions have been done before 45 generations for both variable lengths.

The difference of convergence rate between VIGA-2D and SGA shows that the designed graphical interface gives a suitable understanding to the users for the distribution of the gene values in different generations and to do several interactions. It was noted that for a user, it was a difficult to take decision to make the interactions in the initial 20 generations for the benchmark functions. However, it was observed that after 20 generations, the gene distribution on 2-D graph became easy to understand and to do the interactions. Figure 4.3 (a) shows the visual representation of Rosenbrock's function for generation 10, in which the gene values are distributed in the overall search space. Figure 4.3 (b) shows the gene distribution with 3 interactions and Figure 4.3 (c) shows the gene distribution without interactions after 40 generations. A prominent difference was noticed in the gene distribution with and without interactions for the same generation numbers.

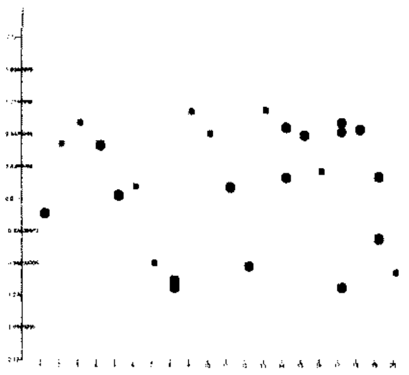


Figure 4.3 (a) : Gene Distribution at Generation 10

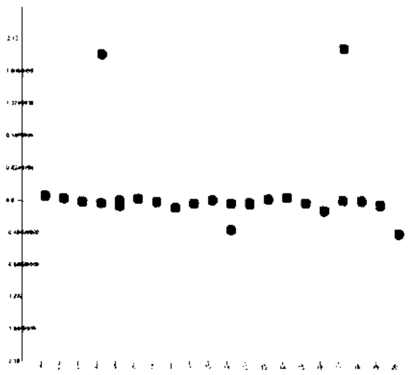


Figure 4.3 (b): Gene Distribution at Generation 40 with VIGA-2D

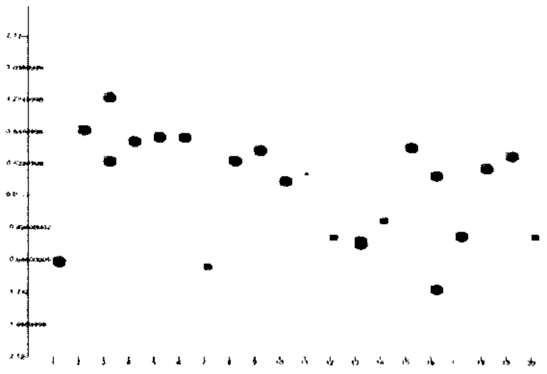


Figure 4.3 (C): Gene Distribution at Generation 40 with SGA

Figure 4.3: Gene Distribution with VIGA-2D for Rosenbrock's Function for n=10.

It was also noticed that the proposed approach has some limitations especially from the human interaction perspective, i.e. lengthy chromosome that may create user fatigue. Therefore the maximum length of a chromosome suggested for benchmark functions was 20. In VIGA-2D, every successful interaction of user will increase population size. Thus, for VIGA-2D the initial population size that was 10 and 20 became 18 and 29 after 100 generation with 8 and 9 successful interactions for n=10 and 20 respectively for Rosenbrock function. Thus this dynamic increment in population size gives a larger search space to the benchmark function interactively. For SGA, the population size was 10 and 20 for all generations. Based on the difference of convergence rate for VIGA-2D and SGA, it was noted that the proposed approach has been proven to be efficient as compared to SGA with both benchmark functions.

4.5 Modelling the Branching Structures (Subjective Analysis)

The application of the proposed approach is to evolve the parameters to derive the rules for the Parametric L-System. The proposed technique for deriving the rules for the Parametric L-System is the further extension of the sketch and grow interface developed for trees [86]. In this existing work, the basic parameters of a drawn sketch are sent to SGA as an optimization problem. After optimization, rules are derived with the L-System symbols retrieved from the initial model sketch by the user and the parameters obtained from best solution of GA. The user may see the next iterations of the growth process in a 3-D space. In the proposed work, the searching ability of GA is used to generate rules for the Parametric L-System with the intervention of the user. Instead of 'sketch and grow' the initial structure, input to VIGA-2D is given by the L-String and the parameters. Furthermore, in proposed approach interactions points or generations for interactions were not initially described to users. Hence the purpose is to evaluate the understanding of users with User Interface and to decide the interactions according to tips given on VIGA-2D tool for understanding gene convergence and to do interaction. On the hand, in proposed approach the interactions are not fixed to some pre-defined parameters. Additionally, the termination of proposed approach is based on user perception.

The best solution obtained from the VIGA-2D is used to model the further growth steps of branching structures. In order to achieve this goal, the proposed approach was evaluated with different GA operators and input L-String. These experiments were based on different users evolved the parameters for deriving the rules for the Parametric L-System. For these experimental results, five participants were invited to run the VIGA-2D with different parameters. These participants were undergraduate and postgraduate students of the ages 22 to 40 years old. All the participants do not have background relevant to the proposed approach; so that the proposed approach should be evaluated on the basis of different users' abilities and perceptions according to their knowledge (see Appendix D for user's background). For example, a few of these users have a good knowledge of the L-system rules and the Parametric L-System but have no knowledge for GA. Moreover, some of the users have no knowledge regarding the rules or the grammar of the L-System, but have a good knowledge about GA. In this way, the resultant branching structures and convergence rate were recorded with different observations, based on the users' skills. Experiment demo were given to these users for understanding the nature of the proposed approach. The participants were also informed that they can interact at any generation during the searching process. They were also informed about the different colours used in visualization for understanding the searching process.

Initially, every participant runs VIGA-2D with different operators to explore different output based on different selection and mutation and crossover rate. However, later on, all users were requested to use common GA operators, selection method and population size to keep the consistency. From among these several experiments with same parameters, one result from each participant was selected to be presented and discussed here. The rest of the results can be found in Appendix D. However, users were allowed to run VIGA-2D with different chromosome lengths and numbers of generation. In regards to this, the following are some assumptions and value which remained constant for all experiments.

- The axiom is predefined. For every experiment the axiom is = (0.24, 0.6,0)A(x,w), where x and w are optimizes using VIGA-2D.
- The chromosome length for the parameters was dynamic, based on the user' input for the symbols.
- The input string may have different lengths with the maximum of 15 symbols.

The approximated time for each process was from ten minutes to half an hour for *Experiment 1*. For *Experiment 2*, it was 5 minutes due to the constant number of generations.

The process was initiated with the user input string. This string was composed of the different symbols used to generate the initial branching structures. The initial parameters were also input by the user. This input string and the parameters help to generate a target solution for VIGA-2D by generating random values. The length of the random chromosome and the initial population of VIGA-2D were based on the length of the user input string (see Section 3.5.2). For the input chromosome, different gene locations have different ranges of values. During the searching process, the user may interact several times, propose new gene values and model the structure at any generation. For modelling the branching structures two modules *'F'* and *'A'* are used, where *'F'* is responsible for producing the line and *'A'* is used as the interpretation point. The following are the different experiments which have been done with the proposed approach.

- Experiment 1: Evaluate the performance of VIGA-2D according to human perception and comparison with SGA.
- Experiment 2: Generate the structures with a constant generation number.

Each experiment has been done with 5 participants. Each subjective result is named by "Result", for example, Result 1 belongs to User 1. The overall performance of the process was observed through 2-D graph for gene evolution in different generations, fitness convergence and generated branching structures. Except for the generation number, which depends on the termination done by the user and level of iterations in Experiment 1, all other GA parameters were the same for all experimental results which are as follows:

- Roulette Wheel Selection method was used.
- One point crossover was performed.
- Mutation was performed with a mutation rate 0.05.
- Initial population size was 10.

4.5.1 Experiment 1: Evaluate the Performance of VIGA-2D According to Human Perception and Comparison with SGA.

The objective of this experiment is to terminate the searching process according to human perception after generating the branching structures in different generations. The purpose of this experiment is to evaluate and observe the human understanding with gene distribution on 2-D Graph and human interventions for generating the branching structures with VIGA-2D. VIGA-2D was evolved up to a number of generations according to the user or until there was no further improvement in the solutions. The user may model the branching structure at any generation. The results presented here are the branching structures modelled by the user at different generation numbers.

In literature survey done in chapter 2, it was noted that in the traditional IGA, the optimized results were based on user perception, in which users play an important role and provide a means of subjective analysis [15, 17]. Most of the existing IGA techniques were based on the user interactions in each generation for assigning the fitness and for selecting the parents for the next generation. This continues interaction of user often creates a tiresome environment for the user [15]. In addition, it was also noted that in existing techniques for IGA, a numerous work has been done to reduce the user fatigue. However, in these existing research works, it was noticed that a learning mechanism was introduced to assign the fitness to the individuals [44, 45]. For example, Takagi uses a discrete fitness value [25] to assign the fitness. In [44, 45] a Neural Network approximation approach has been used for assigning the fitness to individuals. Main drawbacks of these existing methods were to assign fitness values with some fixed parameters and user involvement was also compulsory to select parents for the next generation. Hence, there was no any existing technique, in which, the searching process of GA can be accelerated by human interventions without continues user involvement in each generation.

Several existing techniques had been noticed in literature reviews to model the branching structures using SGA [33, 84, 85]. Besides using SGA, Roger Curry [33] used an interactive approach of GA in which, assigning of fitness was in every generation. In their work, the objective was on generating the next generation with mutation or crossover between parents. The decision for selecting parents for the next generation was on the basis of user perception.

The proposed approach is based on the human-machine interaction. The contribution of the user is to propose a new individual into the current generation instead of evaluating the existing solutions. The purpose of proposing new solutions through interaction is to accelerate the performance of the GA searching ability. Moreover, the interaction of the user was not necessary in each generation.

In order to evaluate the performance of the proposed approach with user interaction, experiments were carried out to compare the results with and without user interaction during the search space. These experiments were based on generating the branching structures with SGA and VIGA-2D. All the operators, selection methods and parameters used to evolve SGA were the same as in VIGA-2D.The level of iteration number for generating structures was 2 for both algorithms.

Result 1

Input string by user*: !F[+]F[-]*

***Input parameters** : 20, 3,26*

***L-String** : !F[+!F[+]F[-]]F[-!F[+]F[-]]*

Input to VIGA-2D

Random Generated Chromosome (n=21):

20,3,26,5,30,16,29,27,30,7,15,34,12,15,8,12,30,32,8,6,2

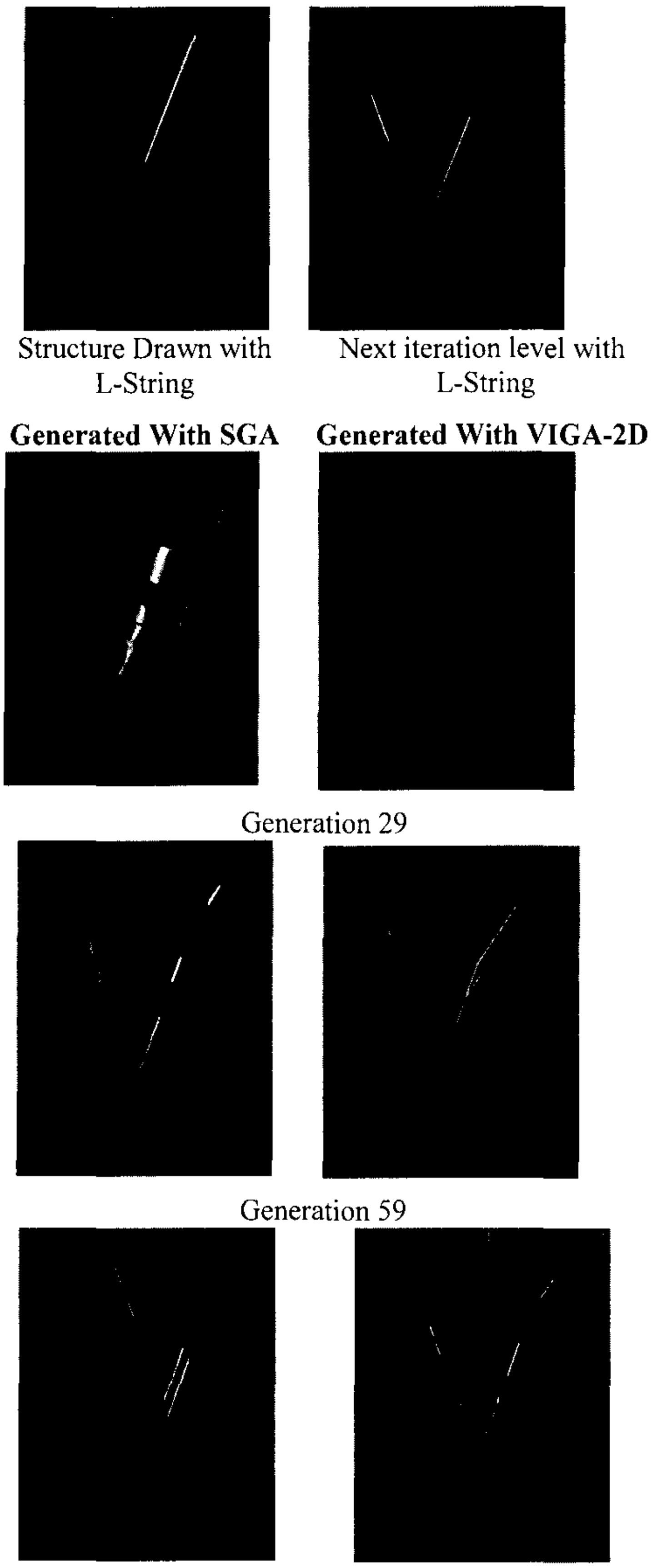


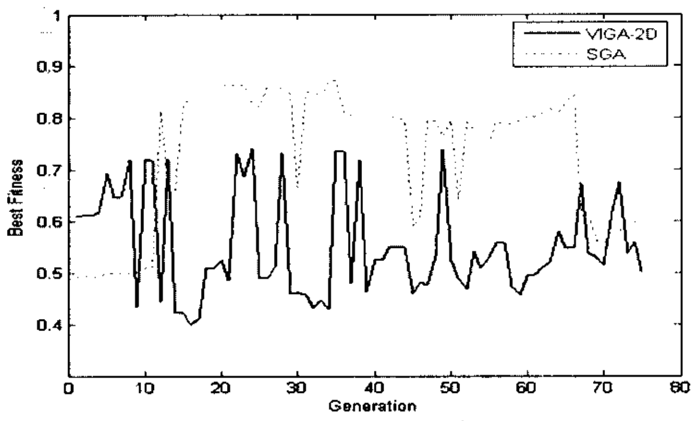
Figure 4.4: Result 1: Branching Structures with VIGA-2D and SGA

Figure 4.4 shows generated branching structures with and without user interactions. In this experimental result the branching structures were generated at 29, 59 and 75 generations. Rule derived for this experimental result in generation 29, 59 and 75 shows a variation in parameters for SGA and VIGA-2D. It was observed that the in generation 29, both algorithms had not obtained a satisfactory solution. Although for VIGA-2D 4 interactions had been made before generation 29. A difference was noted in generation 59 between both algorithms for generated branching structures. In generation 75 VIGA-2D was successful in creating an acceptable structure as compared to SGA. The graph of fitness / generation in Figure 4.5 shows the difference in the convergence of fitness between SGA and VIGA-2D.

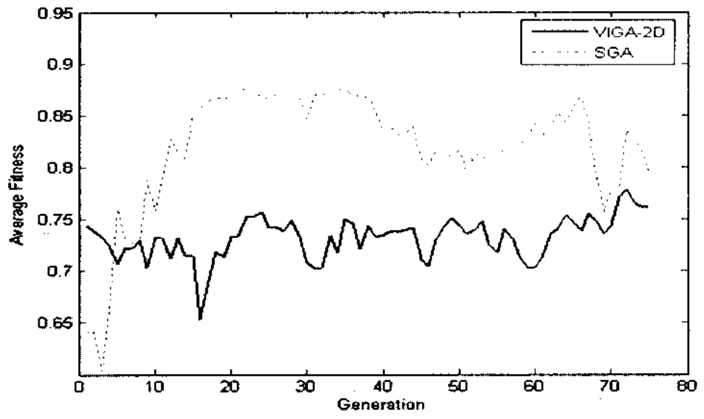
In total, 14 interactions had been made by the user in 75 generations. From 14 interactions, 3 proposed individuals were discarded during the searching process as shown in Table 4.5. However, there was not any significant difference noted for generations 59 to 75 for derived rule and generated branching structures with VIGA-2D. The discarded proposed solutions were also in the same phase. The user decided to terminate the process at generation 75 by concluding that there was no any major difference between generated structures after generation 60.

Table 4.5: Result 1: List of Accepted and Discarded Proposed Individuals at Different Generations

Model Generated at Generation	Interactions at Generations	Best Fitness for VIGA-2D	Best Fitness for SGA	Discarded at Generation
29	11,16,21,27	0.459565	0.8459905	Null
59	33,38,42,46,50,58	0.456618	0.79642953	38
75	60,67,72,74	0.50019	0.5998333	67,72



Best Fitness/ Generations



Average Fitness/ Generations

Figure 4.5: Fitness/Generation Graph for Result 1

Result 2

Input string by user*:* !F[-]F[+]F

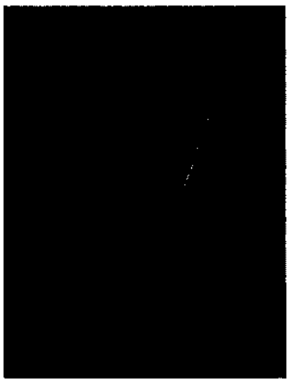
Input parameters ***:*** 27, 3, 21, 19

L-String *: !F[-!F[-]F[+]F]F[+!F[-]F[+]F]F*

Input to VIGA-2D

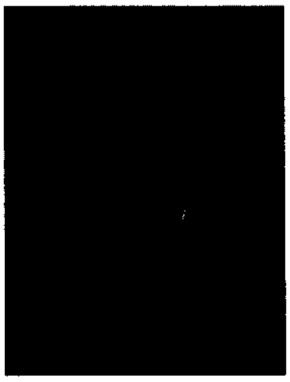
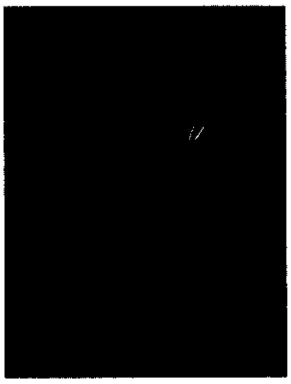
Randomly Generated Chromosome (n=24):

27,3,21,19,18,6,12,22,13,25,17,21,26,32,11,31,32,33,22,20,16,15,9,20

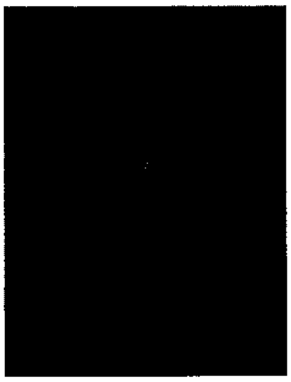


Structure Drawn with L-String Next iteration level with L-String

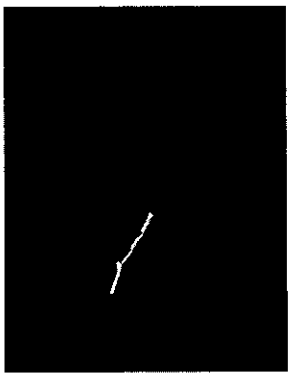
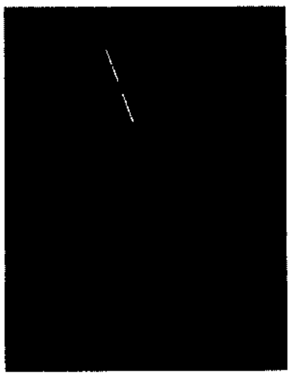
Generated With SGA **Generated With VIGA-2D**

Generation 30



Generation 51

Generation 78

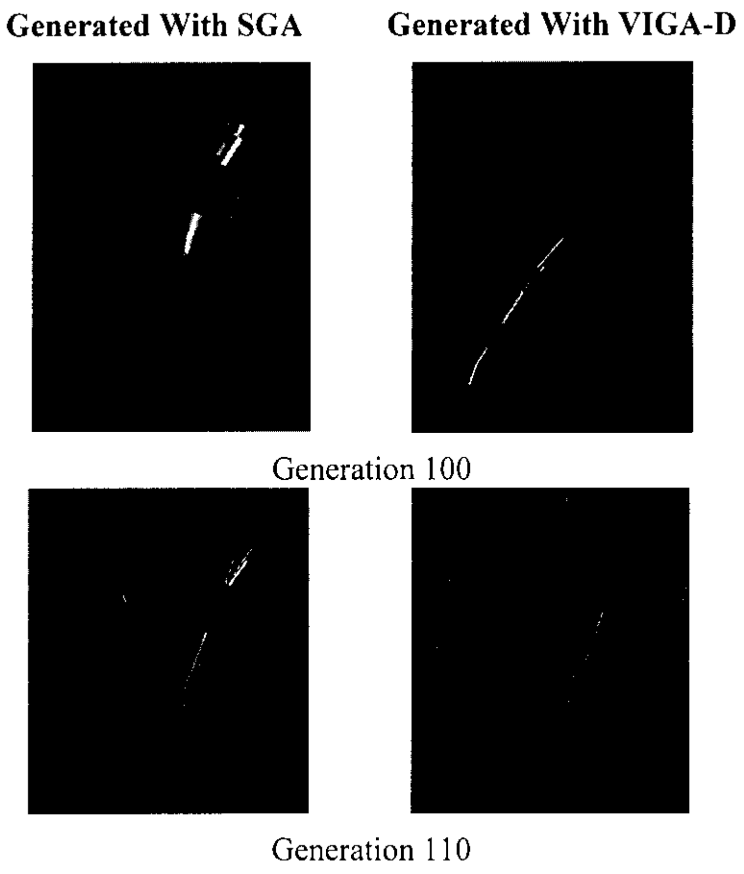
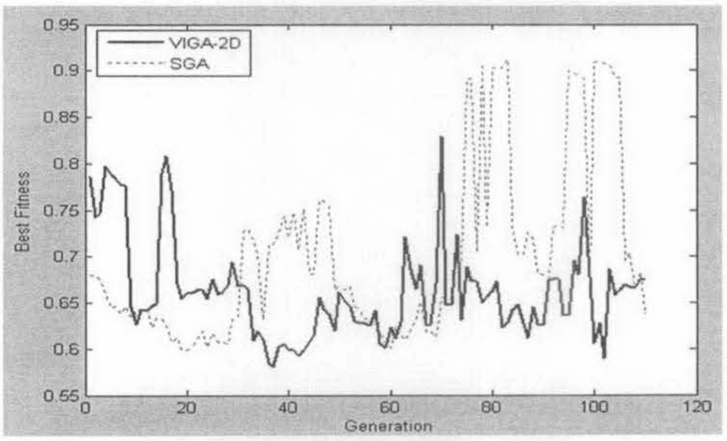


Figure 4.6: Result 2: Branching Structures with VIGA-2D and SGA

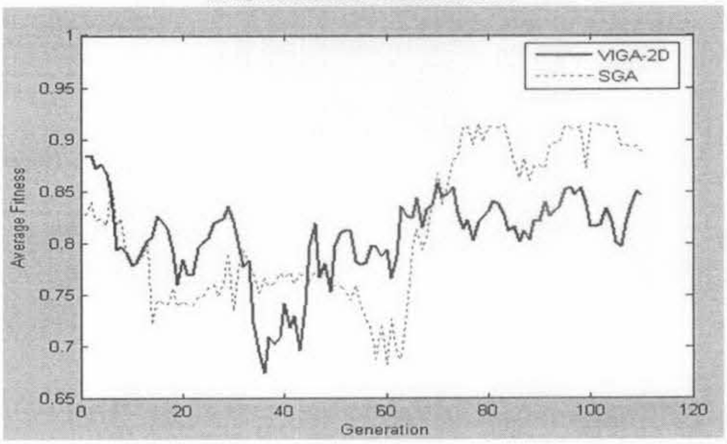
The structures drawn in Figure 4.6 show the results for the searching process until 110 generations, with 24 chromosome length for both algorithms. The branching structures were generated at generation numbers 30, 51, and 78, 100 and 110. The branching structures generated in generations 51, 78 and 100 shows a difference between SGA and VIGA-2D. However, there was not a prominent difference between the branching structure generated in generation 110 with SGA and VIGA-2D. The interaction Table 4.6 shows that the users did a number of interactions and continuously modelled the structure to bring an acceptable output according to his perception. From the analysis of user interactions, it is found that 4 proposed solutions were discarded by VIGA-2D for this experiment. The first structure was drawn at the generation number 30, and the population size was 13 with 3 interactions. In total, there were 15 interactions. At the time of termination the population size was 22. The fitness / generation graph in Figure 4.7 shows the difference between the convergence of SGA and VIGA-2D for this experimental result. The results show that the structures drawn in generation 30 and 110 have the closest parameter values for derived rule. The user decided to terminate the searching processing at generation 110.

Table 4.6: Result 2: List of Accepted and Discarded Proposed Individuals at Different Generations

Modelled at Generation	Interactions at Generations	Best Fitness for VIGA-2D	Best Fitness for SGA	Discarded at Generation
30	13,19,25	0.668730	0.634901	Null
51	34,43,50	0.652150	0.664003	43
78	60,73,77	0.65	0.907469	Null
100	83,89,94,99	0.605640	0.908350	89,94
110	105,109	0.67453687	0.6385139	Null



Best Fitness/Generation



Average Fitness/Generation

Figure 4.7: Fitness/Generation Graph for Result 2.

Result 3

Input string by user: *!F[[-F]]F[+F]*

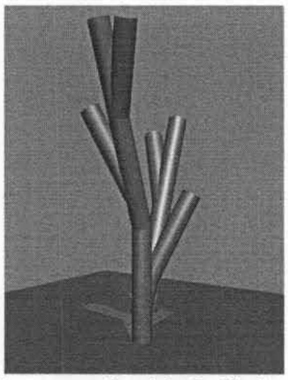
Input parameters *: 10, 7, 34, 28, 30, 31*

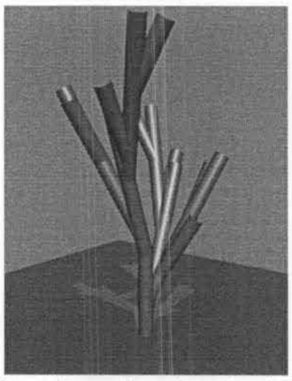
L-String *: !F[[-!F[[-F]]F[+F]F]]F[+!F[[-F]]F[+F]F]*

VIGA-2D

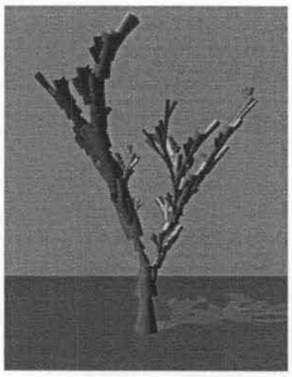
Randomly Generated Chromosome (n= 27):

10,7,34,28,30,15,13,13,29,21,23,30,14,34,20,19,17,22,25,31,13,9,11,1
9,15,5,11

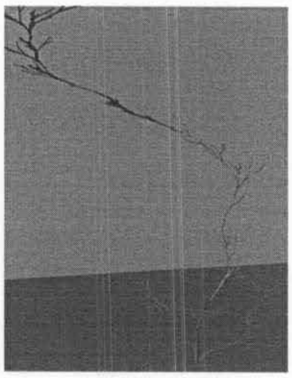

Generated with L-String


Next iteration level with L-String

Generated With SGA



Generated With VIGA-2D



Generation 31

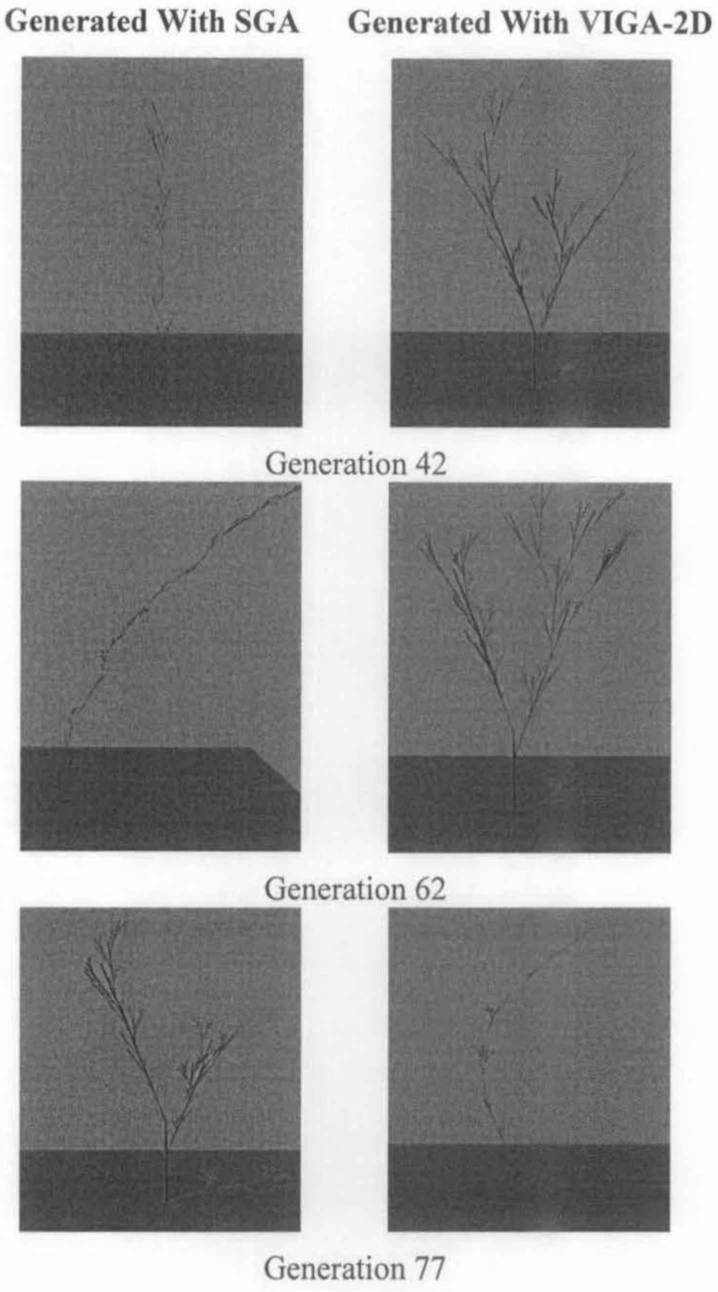
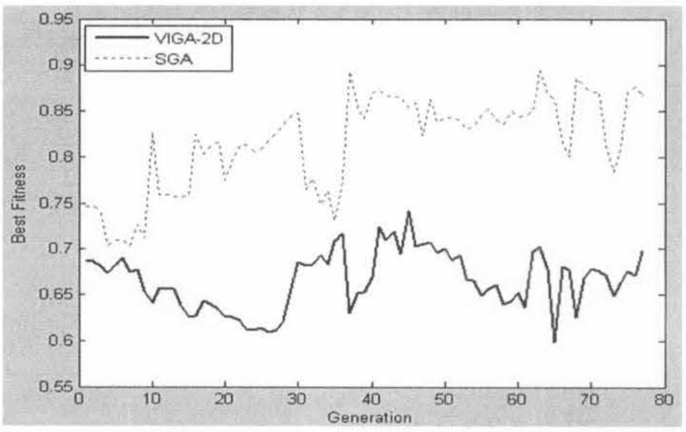


Figure 4.8: Result 3: Branching Structures with VIGA-2D and SGA

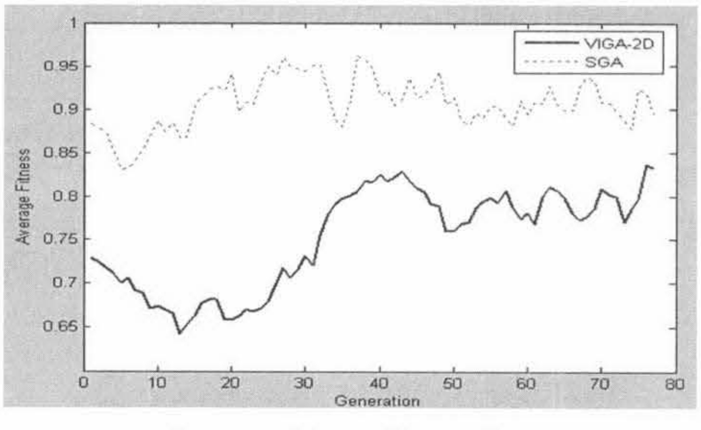
In Figure 4.8, branching structures were generated after 31, 42, 62 and 77 generations. The branching structures and derived rule in 42 and 62 generations shows a difference in parameters with VIGA-2D as compared to SGA. Table 4.7 shows the best fitness values in both generations while modelling the structure. In total, 77 generations were evolved with the chromosome length of 27. For VIGA-2D, 13 interactions were done from which 4 proposed solutions were not accepted by VIGA-2D. The convergence rate of VIGA-2D was noticed to be at optimal fitness values at all recorded generations as compared to SGA The graph of fitness / generation in Figure 4.9 shows the difference of the best and the average fitness for both algorithms.

Table 4.7: Result 3: List of Accepted and Discarded Proposed Individuals at Different Generations

Modelled at Generation	Interactions at Generations	Best Fitness for VIGA-2D	Best Fitness for SGA	Discarded at Generation
31	11,15,19,23,27,	0.681395	0.7645259	19,23
42	32,37,40	0.709436	0.8661985	37
62	47,50,59	0.6972804	0.8498823	50
77	65,74	0.6977822	0.8677557	Null



Best Fitness/Generation



Average Fitness/Generation

Figure 4.9: Fitness/ Generation Graph for Result 3

Result 4

Input string by user: !F[-F[+F]F][-F]

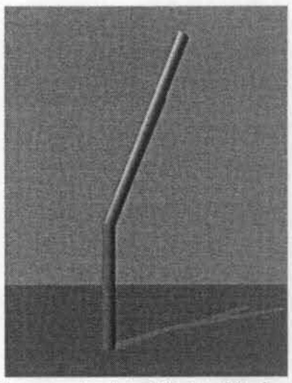
Input parameters *: 27, 3, 34, 11, 22, 33*

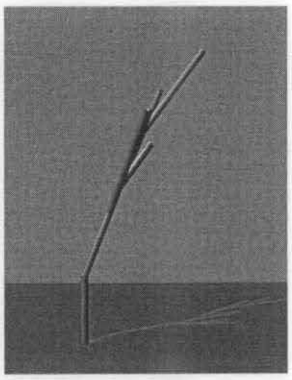
L-String *: !F[-F[+F]F][-F]*

VIGA-2D

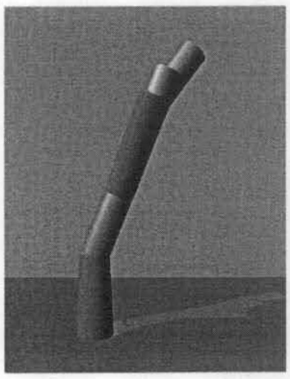
Randomly Generated Chromosome (n= 12):

27,3,34,11,22,33,17,28,5,32,6,14

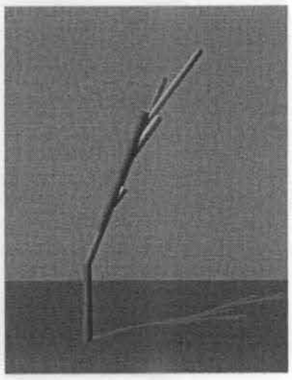

Generated with L-String


Next iteration level with L-String

Generated With SGA



Generated With VIGA-2D



Generation 19

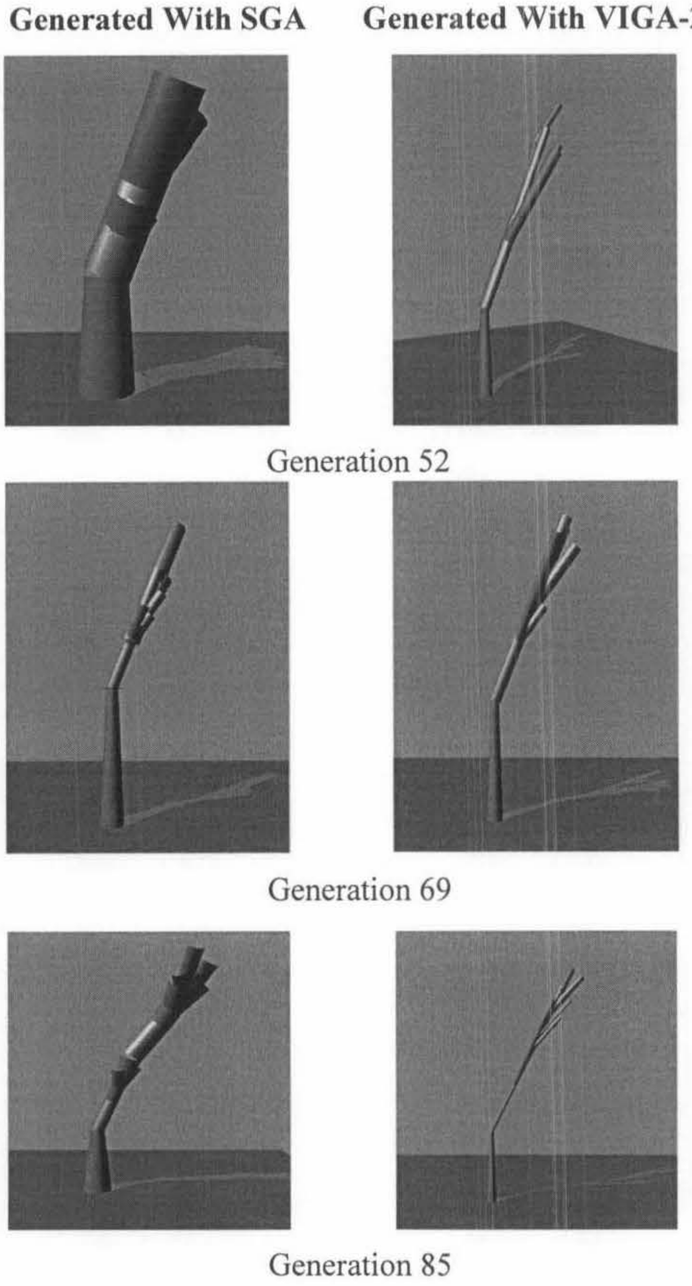
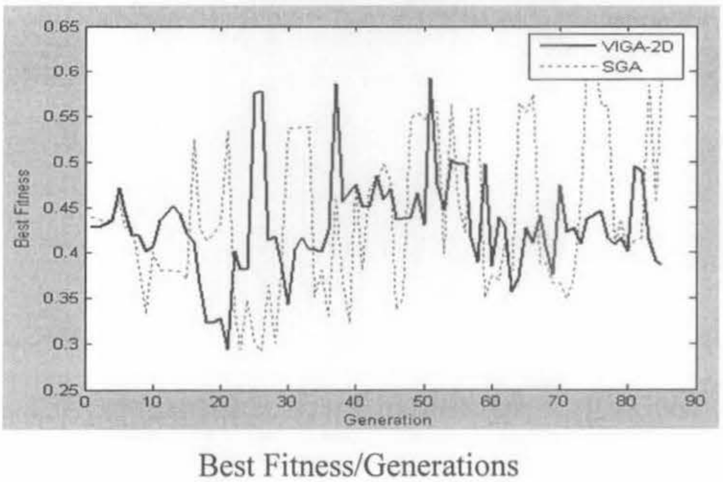


Figure 4.10: Result 4. Branching Structures with VIGA-2D and SGA

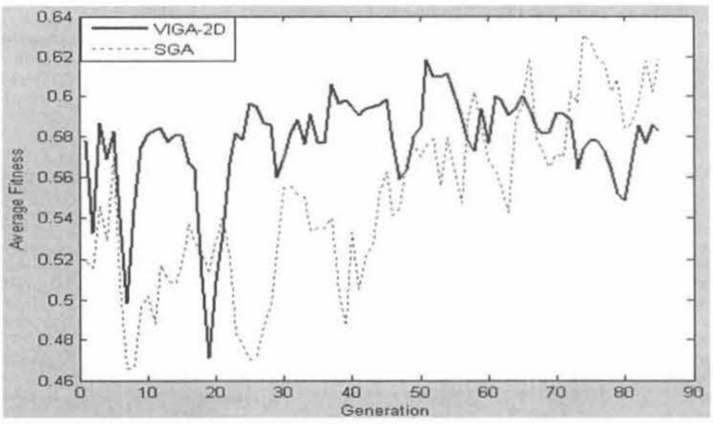
Figure 4.10 shows another result comparison between branching structures generated with and without user interaction. The first structure was generated at generation 19. Both SGA and VIGA-2D were evolved until 85 generations. The derive rules and generated branching structures in generations 19, 52, 69 and 85 show a difference between parametric values for both algorithms. In total, 14 interactions had been done for VIGA-2D from which, 2 proposed individuals were discarded. It was noted that the structures generated at generation 69 by both algorithms have a minor difference between fitness as shown in Table 4.8. The user continues with the next generations and generated another structure at generation 85, and in generating the structure with SGA at generation 85 again the difference in fitness was noted. For VIGA-2D the population size at the termination point was 22. The fitness / generation graph in Figure 4.11 shows the convergence difference for SGA and VIGA-2D for this experiment. According to the fitness / generation graph, the convergence of SGA and VIGA-2D was noted to have the same fitness values in the initial generations. It was also noted that for VIGA-2D the average fitness was higher as compared to SGA for a few generations. However, in the last 20 generations a difference was noted between both algorithms' fitness values. Hence, continues interactions make VIGA-2D to converge at optimal fitness as compares to SGA. Hence, this convergence difference successful to bring an optimized solution at the generation 85 with VIGA-2D.

Table 4.8: Result 4: List of Accepted and Discarded Proposed Individuals at Different Generations

Modelled at Generation	Interaction at Generation	Best Fitness for VIGA-2D	Best Fitness for SGA	Discarded at Generation
19	8, 16	0.32311	0.419880936	16
52	28,32,42,49	0.474974	0.554346462	Null
69	57,61,66, 69	0.375366	0.366878727	61
85	72,77,82,85	0.386652	0.589236795	Null



Best Fitness/Generations



Average Fitness/Generations

Figure 4.11: Fitness/Generation Graph for Result 4

Result 5

Input string by user: *!F[F][&]F*

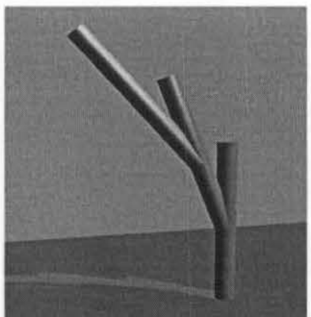
Input parameters *: 12, 5, 10, 15*

L-String *: !F[F][&!F[F][&!F[F][&]F]F]F*

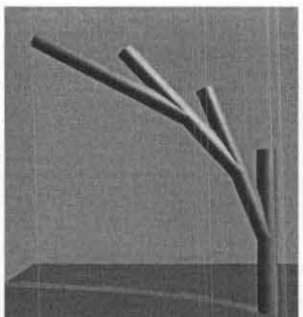
VIGA-2D

Randomly Generated Chromosome (n= 18):

12,5,10,15,19,29,12,25,19,22,10,19,24,6,19,8,28,33



Generated with L-String



Next iteration level with L-String

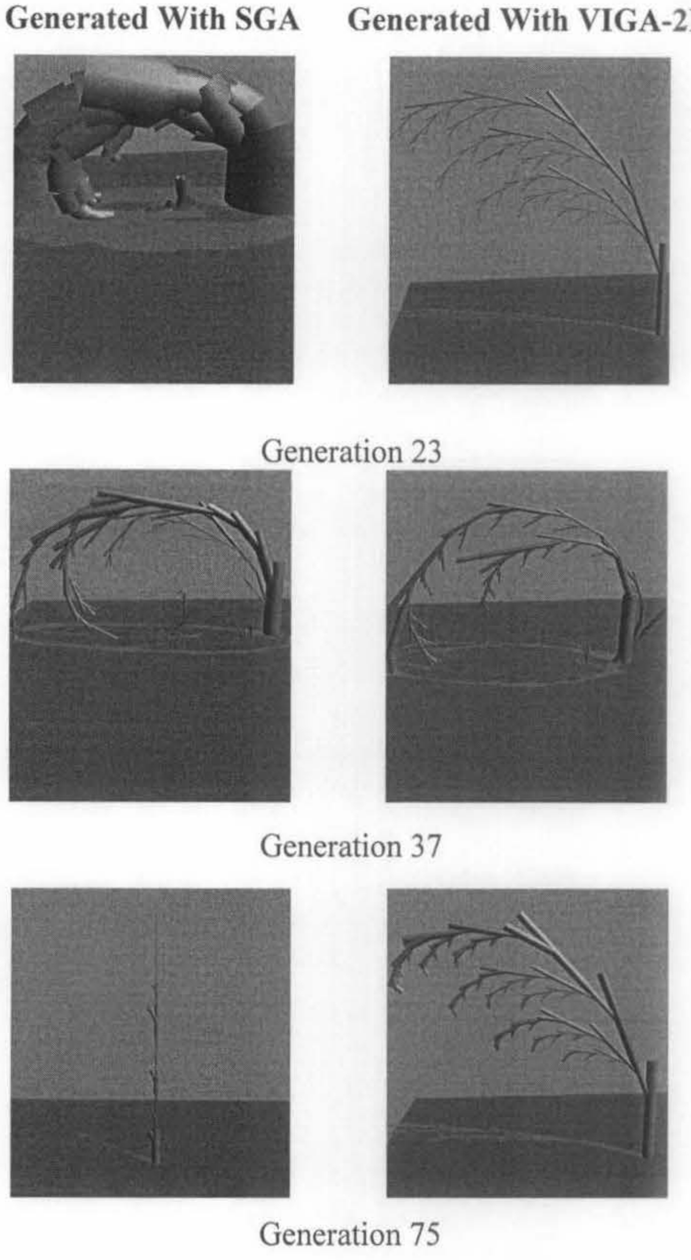
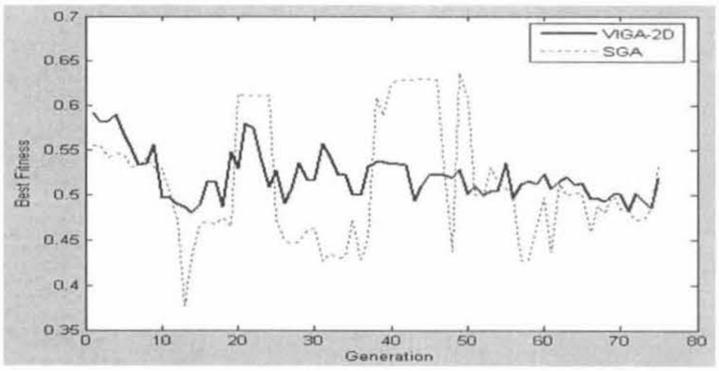


Figure 4.12: Result 5. Branching Structures with VIGA-2D and SGA

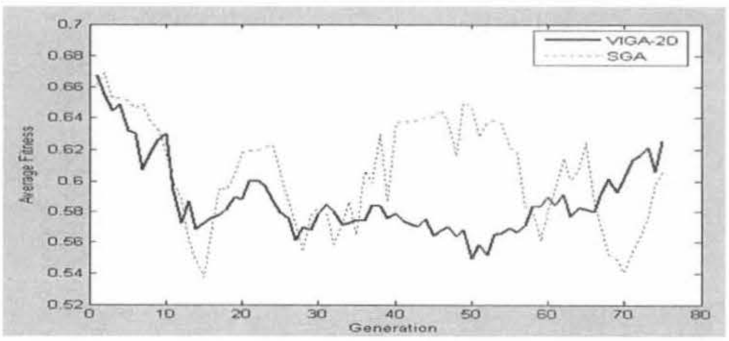
The result generated in Figure 4.12 shows the branching structures until 75 generations. For all the generated structures VIGA-2D shows efficient results as compared to SGA. The branching structures generated in generation 23, 37 and 75 show a difference between SGA and VIGA-2D. A total 19 interactions were done, from which 3 proposed individuals were discarded by VIGA-2D. Hence, at the time of termination the population size was 19. Table 4.9 shows the generations which were interrupted by the users' for VIGA-2D. Figure 4.13 shows the fitness convergence graph for this result.

Table 4.9: Result 5: List of Accepted and Discarded Proposed Individuals at Different Generations

Modelled at Generation	Interactions at Generations	Best Fitness for VIGA-2D	Best Fitness for SGA	Discarded at Generation
23	12,18, 22	0.54147	0.610982	Null
37	25,31,35	0.53197	0.451425	35
75	41,50,57, 61,69,73	0.51980	0.531601	50.61



Best Fitness/Generations



Average Fitness/Generations

Figure 4.13: Fitness / Generation Graph for Result 5

4.5.1.1 Discussion with Experiment 1.

In order to understand the efficiency of the proposed approach with intervention of the user, branching structures were generated with SGA and VIGA-2D. Table 4.10 shows the entire parameter list and the difference of average and best fitness for both algorithms. The users may generate the branching structure in any generation to see the output, so that they can decide to terminate or go to more generations. The decision of termination of process depends on the modelled branching structure as compare to initial model or the convergence rate. Although, the averages and best fitness for both algorithms have a very small difference at the time of termination for most of the results, the generated branching structures with both algorithms seem to have a difference in the angles and different parametric values in different generations. With the comparison of averages and best fitness difference for both algorithms, it was found that the convergence rate for VIGA-2D shows a better performance as compare to SGA. However, for *Result 1* the average fitness of SGA was found to be lower as compared to SGA with the difference of 0.0342. For *Result 2*, the user evolved 110 generations, did not successfully bring the VIGA-2D to a better convergence rate and the difference for the average fitness of both algorithms was found to be -0.*043196*. For *Result 3*, the average fitness difference for both algorithms was -0.*0633*, for Result 4 it was -0.*03507* and for *Result 5* it was 0.0197. Table 4.10 also shows all the details of the parameters taken for this experiment. According to the Table 4.10, the total number of interactions noted for all the results were close to each other; whereas, the number of generations evolved by all the users was different. Overall, the performance of VIGA-2D shows a prominent difference while generating the branching structures as compared to SGA.

Table 4.10: List of Parameters used for Different Experimental Results in Experiment 1

Parameters	Result 1		Result 2		Result 3		Result 4		Result 5	
	VIGA-2D	SGA	VIGA-2D	SGA	VIGA-2D	SGA	VIGA-2D	SGA	VIGA-2D	SGA
Population Size	10	10	10	10	10	10	10	10	10	10
No of Generations	75	75	110	110	77	77	85	85	75	75
Total Interactions	14	N/A	15	N/A	13	N/A	14	N/A	12	N/A
Accepted Interactions	11	N/A	12	N/A	09	N/A	12	N/A	09	N/A
Discard Interactions	03	N/A	03	N/A	04	N/A	02	N/A	03	N/A
Population size after Interactions	21	10	22	10	19	10	22	10	19	10
Average Fitness	0.76179	0.79590	0.8455	0.88869	0.8319	0.8943	0.58303	0.61818	0.62538	0.60567
Best Fitness	0.50019	0.59983	0.67453	0.63851	0.6977	0.8677	0.38665	0.58923	0.5198	0.53160

4.5.2 Experiment 2: Generating Structures with Constant Generation Number

For the traditional IGA applications, the applications were based on the constant number of generations, in which the interaction of the user was necessary in each generation for evaluating and assigning the fitness to the solutions [21, 33]. The proposed approach is different from these applications and does not require user interaction in each generation. The fitness of the solutions was calculated with the objective function and there is no need to evaluate the solutions for the next generation.

Experiment 2 is based on constant number of generations and interactions in order to evaluate the performance of the proposed approach with different users. For this experiment input string and initial parameters were same for all users. The purpose of this experiment was to monitor the behaviour of VIGA-2D with fixed parameters. This experiment also helped to prove the random searching ability of GA. The main feature of this searching technique is to generate different solutions, even with the same input. All 5 users participated in this experiment. The following are the rate and operators that remain constant for all users for this experiment:

- Number of generations evolved = 40
- Number of interactions =05

Input by User: !F[[-F]]F[+F]

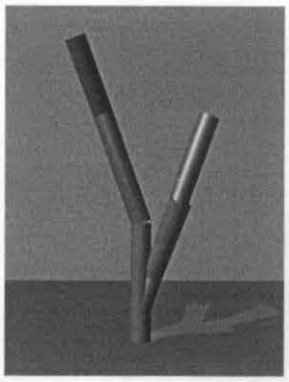
Input Parameters: *11, 7, 29, 28, 26, 7*

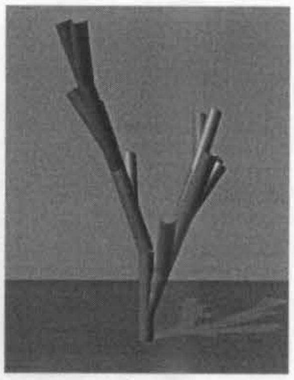
L-String : !F[[-!F[[-F]]F[+F]F]]F[+!F[[-F]]F[+F]F]

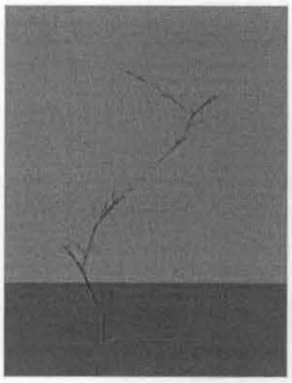
<u>VIGA-2D for all results</u>

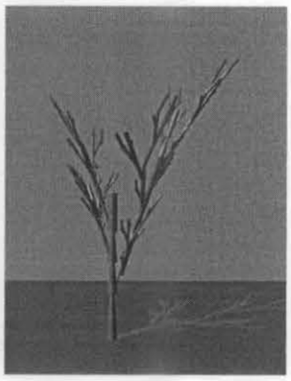
Randomly Generated Chromosome (n=27):

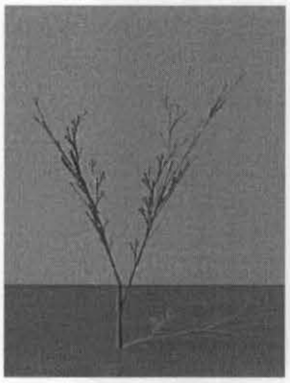
11,7,29,28,26,23,22,23,32,12,5,32,30,31,33,23,28,10,29,23,29,12,22,3 3,25,34,29

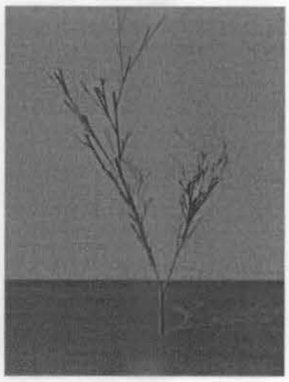

Structure drawn with L-String


Next Iteration Level with L-String


Result1


Result2


Result 3


Result 4

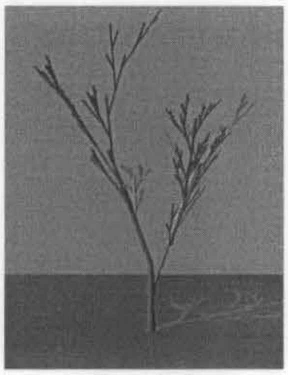
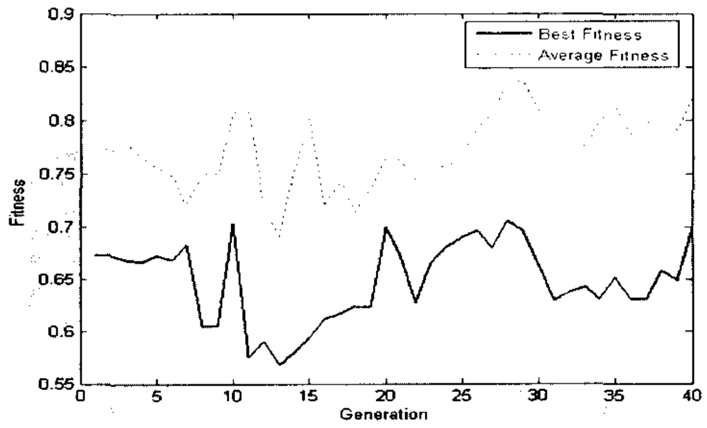

Result 5

Figure 4.14: Branching Structures Generated with Constant Number of Generations.

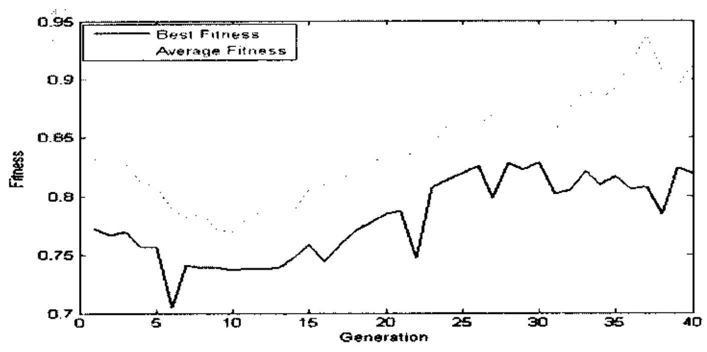
Table 4.11: List of Discarded and Accepted Proposed Individuals at Different Generations for Experiment 2.

Results	Interaction at Generation Numbers	Best Fitness for VIGA-2D	Discarded Interactions
1	12,18,23, 27,33	0.6997	18
2	14,18,22,26,30	0.81865	22,26
3	6,11,18, 24,28	0.62872	6,11
4	10,17,25, 30,35	0.64350	30
5	9,14,19,23,27	0.68505	Null

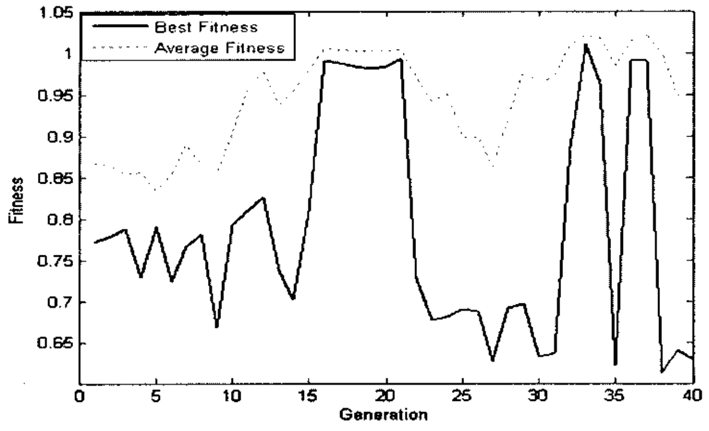
Presented results were based on the fixed parameters, number of generations and iterations. Structures drawn after 40 generations with all users are shown in Figure 4.14. The chromosome length for these experimental results was 27. Table 4.11 shows the list of generations in which user interacts, and the generations at which the proposed solutions were accepted or rejected. For *Result 2* and 3, two individuals and for *Result 1 and 4*, one proposed individual were rejected with VIGA-2D. Whereas, for *Result 5*all 5 proposed individuals were accepted by the VIGA-2D.



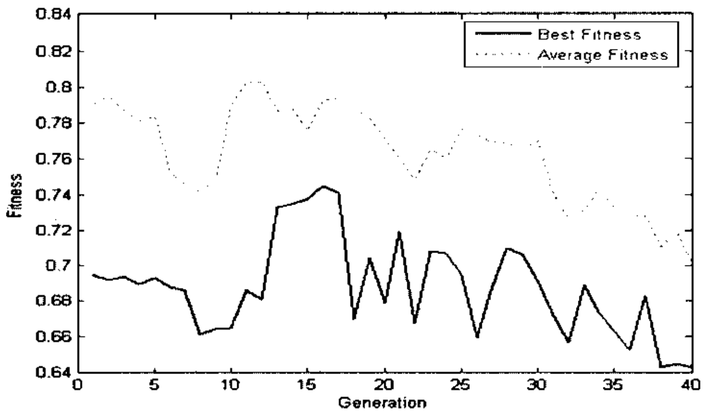
Fitness /Generation Graph for Result1



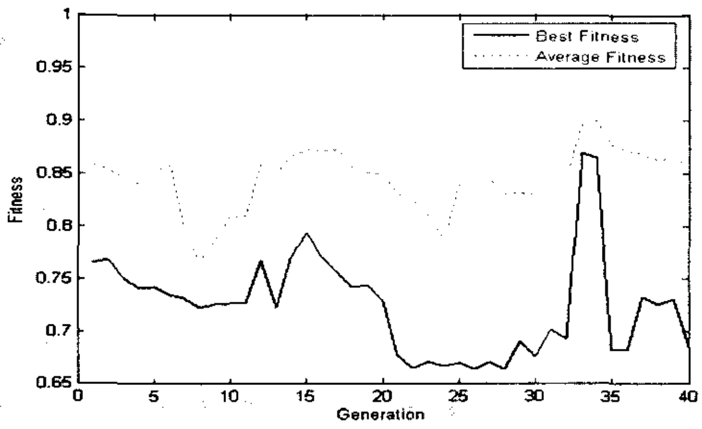
Fitness /Generation Graph for Result2



Fitness /Generation Graph for Result3



Fitness /Generation Graph for Result4



Fitness /Generation Graph for Result5

Figure 4.15: Fitness/Generation Graph with Constant Number of Generations

Table 4.12: List of Parameters used for VIGA-2D Process with Constant Number of Generations

Parameters	Result1	Result 2	Result 3	Result 4	Result 5
Population Size	10	10	10	10	10
No. of Generations	40	40	40	40	40
Total Interactions	5	5	5	5	5
Accepted Interactions	4	3	3	4	5
Discard Interactions	1	2	2	1	0
Population size after Interactions	14	13	13	14	15
Average Fitness	0.8213	0.91276	0.92646	0.7009	0.85520
Best Fitness	0.6997	0.81865	0.62872	0.64350	0.68505

4.5.2.1 Discussion with Experiment 2

The understanding of users with fixed parameters was observed in Experiment 2. Table 4.12 shows all the parameters and effect on these parameters with user interactions. Table 4.12 shows that after 40 generations all the experimental results converged at closely same best fitness value except result 2. However, the average fitness value for all experimental results shows a difference from each other.

It was observed that fixed parameter for the interactions and generations create confusion to most of the user to make a decision for interactions. While running the VIGA-2D, most of the time the users were observed while they decided on the generations for interactions. However, a few users were observed to be more comfortable as the generations and interactions were already decided on. Figure 4.15 shows the fitness/generation graph for all experimental results. The generated structures with Experiment 2 also show a clear difference in derived rule with different users.

4.6 Advantages of using Visual Aspects of VIGA-2D

GA is often known as a blind search method [5, 6] because it does not require any information about the first derivative or any other restrictive assumption before solving a problem. Unlike other techniques of AI, GA is more robust (error free), even in the presence of small noise or any small change in the input; it does not break easily. The searching mechanism of this algorithm works differently with different selection methods, operators and input. In past, the graphical representation of GA was based on either the individuals of each generation [34] or a representation of a complete population or generation in the form of graphs or plots [57-60]. Using these existing techniques, the gene distribution of the GA process remains hidden from the user. Moreover, these existing visualization techniques for GA do not facilitate analysis of the change in the searching behaviour of GA according to the change in operators, selection methods or in chromosome.

The main idea of the proposed research is the representation of multidimensional data on a 2-D graph for each generation. Presentation of the GA process on a 2-D graph makes it easy for a user to evaluate the gene distributions at different places. The experiment was done to elaborate the advantage of using the proposed visualization technique for monitoring the general behaviour of the GA searching process. For this purpose, different selection methods and operators are used to evaluate and discuss the performance and effectiveness of the proposed GUI. In this experiment, the GA process was evolved until different numbers of generations to monitor the gene distributions. The length of input chromosome was 24 for all experimental results and population size was 10. The discussion and analysis of the following results are based on the genes distributions in different generations.

4.6.1 Roulette Wheel Selection Method, 1-Point Crossover and Mutation Rate 0.05.

The experimental results in Figure 4.16 show the gene distributions for parameters at generation 1, 20, 40 and 60. This experiment had been done with roulette wheel selection method, 1-point crossover and with mutation rate was 0.05. In these results, the noted difference was in the first location of gene values. The figure 4.16 (a) shows the gene distribution in the first generation. It was noticed that initially gene values were exists at the first gene location in first generation. However, with the evolving of more generations less gene values distribution were noted at first gene locations (see Figure 4.16 (b)). After the 20th generation, there were 2 interactions by the user and the distribution of gene values improved in 40 and 60 generations. Overall, gene distributions at different generations with roulette wheel selections were equally distributed. It was noted that with roulette wheel selection method, the gene values were fairly distributed in different generations.

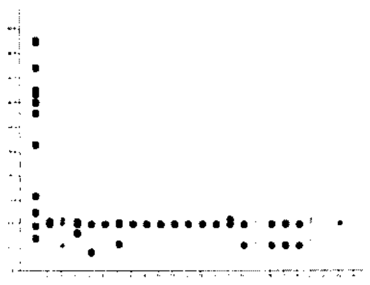


Figure 4.16 (a) Generation 1

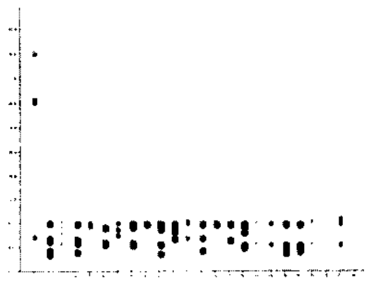


Figure 4.16 (b) Generation 20

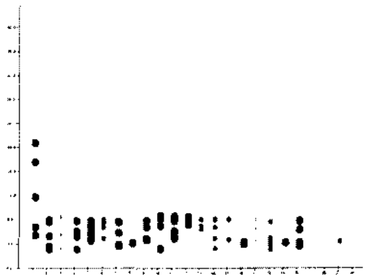


Figure 4.16 (c) Generation 40

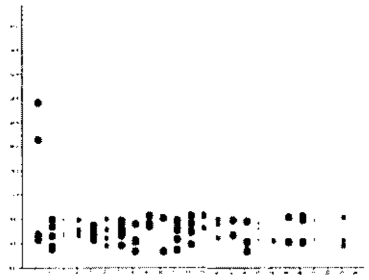


Figure 4.16 (d) Generation 60

Figure 4.16: VIGA-2D with Roulette wheel selection.

4.6.2 Tournament Selection Method, 1-Point Crossover and Mutation Rate 0.05

For results in Figure 4.17, tournament selection was used. The mutation rate was 0.05 and 1–point crossover was used. It was observes that, the gene distribution at different generations with tournament selection were not equally distributed as compared to the roulette selection method. Although in the generation 40 (see Figure 4.17 (c)) there was an improvement in the gene distribution, in the generation 60 the 2-D Graph shows again a poor gene distribution. The interactions were done in generation 18, 22, 32 and 46. Different experiments show that for deriving rules for the Parametric L-System, roulette wheel selection was more efficient as compared to tournament selection. This difference is noted because of unknown direction of gene values, which were randomly generated to initiate the process. Another reason is in the fact that for tournament selection, fitter individuals are selected only one time for matting pool [105], however for roulette wheel selection method a fitter individual may be selected several times based on its cumulative probability.

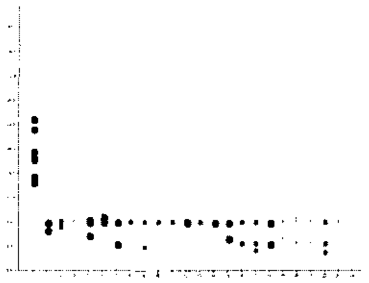


Figure 4.17 (a) Generation 1

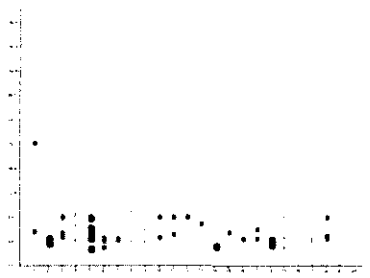


Figure 4.17 (b) Generation 20

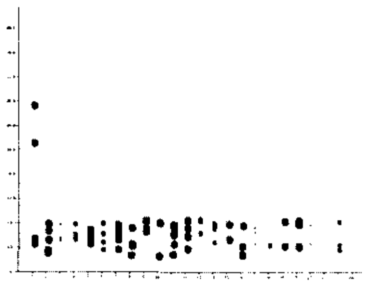


Figure 4.17 (c) Generation 40

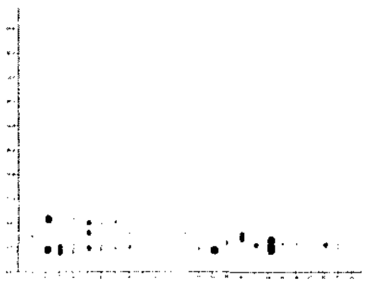


Figure 4.17 (d) Generation 60

Figure 4.17: VIGA-2D with Tournament Selection

4.6.3 Mutation Rate 0.5, 1-Point Crossover and Roulette Wheel Selection Method

Mutation rate is the most important and sensitive operator used to control the optimal convergence [116,117]. For example a higher mutation rate may lead loss of the potential solutions [115] or a smaller mutation rate may not able to give the desired output. An optimal rate for mutation is an important part to get an optimal solution [116, 118]. In other sense, the performance of GA highly depends on the mutation rate. The decision of selecting optimal mutation rate depends on nature of problem. However in proposed approach the difference of distribution of gene values with assigning different mutation rate can be easily monitored using VIGA-2D.

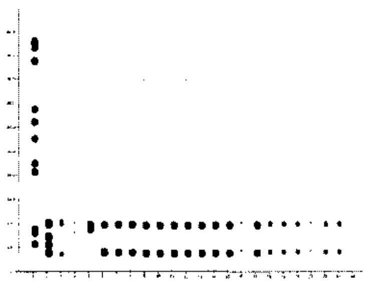


Figure 4.18 (a) Generation 1

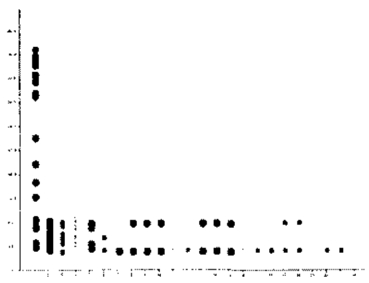


Figure 4.18 (b) Generation 20

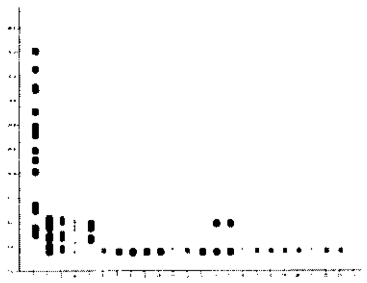


Figure 4.18 (c) Generation 40

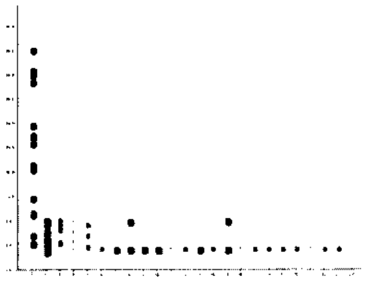


Figure 4.18 (d) Generation 60

Figure 4.18: VIGA-2D with High Mutation Rate = 0.5

A few experiments have been done to see and analysis the gene distribution based on different mutation rates. From these experiments it has been concluded that smaller mutation rate helps to generate an optimal solution for evolving parameters and benchmark functions presented in this thesis. Figure 4.18 shows the results with gene distributions with high mutation rate. For this result, the roulette wheel selection method with 1-point crossover was used. In the first generation (see Figure 4.18 (a)), the entire gene values were found at the same gene location for each gene value. After the generation 20, there was a distribution of gene values to some other places. It was noted in Figure 4.18 (d) that in generation 40, most of the gene locations had not an optimal gene distribution. Graphical presentation of gene values had given a very clear picture of the poor performance of GA with a higher mutation rate. Hence using VIGA-2D, it was noted that the high mutation rate can lead to less equal distribution of gene values in the search space. In turn, it helps in preventing the searching ability of GA to bring the optimized solution in less time or generations.

4.7 Discussion and User Analysis

GA is the search technique used in computing field to find exact or approximate solutions for optimization and searching problems. Finding optimized solution of GA depends on the nature of the problem. If the problem is based on an integer or a string, then to find the optimized solution it is easier as compared to the problem in which the problem is based on real values. In the proposed research, the problem that was given to GA was to optimize the parameters for L-System rules, which were real values. Therefore, most of the chances were to get the nearest optimized solution as compared to get the optimized solution.

The purpose of developing VIGA-2D was to monitor the distribution of gene values with different frequencies and locations. The key objective of the proposed approach was to accelerate the performance of GA searching by proposing a fitter solution in the search space. For this reason, the understanding of gene distribution of the search space on 2-D Graph play important role for optimal user interactions. During evolving different problems, the performance of VIGA-2D was closely observed. Different experiments with VIGA-2D shows that user interactions for benchmark functions was easier as the searching process goes to higher generations as compared to the parameters evolved for the Parametric L-System. For the benchmark function, in the first 20 generations, the decision of interactions was based on the colour intensity as compared to the higher generations in which all the gene values were distributed at some specific location. It was observed that in the experiment with DeJong's function the human decision for interactions was easy as compared to the Rosenbrock's function because of the slow convergence towards an optimum solution in Rosenbrock's function.

Figure 4.19 shows the gene distribution for DeJong's function. The 2-D graph for SGA and VIGA-2D shows a difference between the gene distributions in different generations. It was noted that the gene distributions for the initial generation was almost the same for both algorithms as shown in Figure 4.19 (a). However, in generation 21, the performance of VIGA-2D with 2 interactions has improved as compared to SGA for the same generation number. For VIGA-2D, the searching process was interacted by user at generation number 16, 21, 34, and 45. In total, 4 interactions were done with the initial population size 20. In generation 45, the difference between genes distributions shows a prominent difference in the distribution of gene values into the search space with the help of user interactions.

Gene Distribution with VIGA-2D **Gene Distribution with SGA**

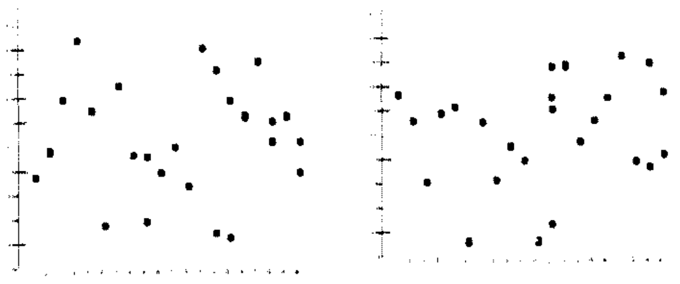


Figure 4.19 (a) Generation 5

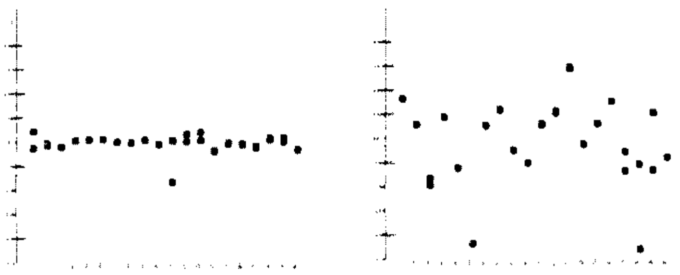


Figure 4.19 (b) Generation 21

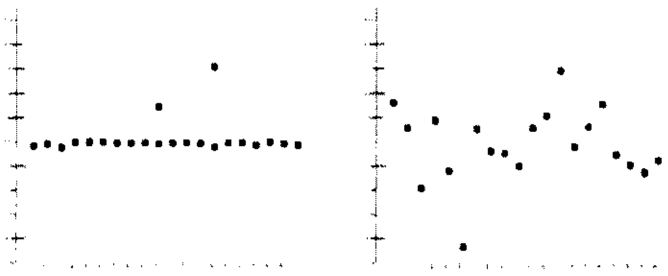


Figure 4.19 (c) Generation 34

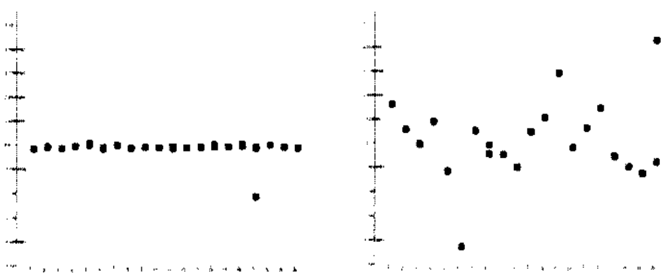


Figure 4.19 (d) Generation 45

Figure 4.19: Gene Distribution with DeJong's Function for VIGA-2D and SGA.

The fitness of parameters for Parametric L-System rule was based on the difference calculated between the target solution (randomly generated) and the GA solution (see Equation 3.14). Hence, a minimum fitness value of individuals leads to bring the optimized solution. Besides the understanding this fitness convergence rate, the gene distributions on the 2-D graph have no specific direction. The distribution of gene values on the 2-D graph can only be monitored by the colour intensities and the size of the gene values. Therefore, while doing the interactions with the search process for optimizing parameters, the user needs to pay more attention to analysing the behaviour of VIGA-2D from generation to generation as compared to the benchmark functions.

While doing the experiment with users for generating the branching structures in *Experiment 1* and *Experiment 2*, the understanding of the user, i.e. observing behaviour with the proposed approach was closely monitored. It should be noted here that the participants of both experiments were the same. Different experiments and observations show that the overall performance and understanding of the users with experiment 1 was more efficient as compared to the experiment 2. One main reason was the use of constant values for interaction and the number of generations for *Experiment 2*. It was observed that the user faced confusion when balancing between generations and interactions. As compared to *Experiment 1*, in which users were not forced to run the VIGA-2D with a constant number of generations and interactions; this gave a more flexible environment to the user in order to bring the desired solution. For *Experiment 1*, it could be difficult to analyse the best results because for every user length of L-String and number of generations was different. Therefore, different convergence rates were noticed for different results as shown in Table 4.10. However, for *Experiment 2*, the different results were based on the same L-String length and number of generations, which brought about a clear picture of understanding and evolving of VIGA-2D with different users.

Although the overall performance and decision for human interaction also depends on a fitness / generation graph, during the interactions, the users were more interested in viewing the output as compared to monitoring the convergence rate. It was observed that in *Experiment 1*, the decision of terminating the searching process were more satisfying to the users as compared to a constant generation numbers, i.e. in *Experiment 2*.

Moreover, it was noted that there was no consistency in doing the interactions. Some users did more interactions in the initial generations as compared to higher generations. It was also noted that some users were more interested in modelling the branching structures as compared to understanding the VIGA-2D environment.

The decision for doing interaction in any generation was a critical point in the proposed algorithm. A continuous observation has been done on the users understanding for the next interactions. These observations are briefly described below:

a) The user interacted in the current generation when there were a lesser number of gene values at a particular place.
b) The gene value's colour intensity was low (towards the gray colour) at any particular gene location.
c) User feels that a group of gene values were at the same gene location from many generations.
d) In VIGA-2D, the propose colour was green which shows that the current generation fitness was not optimal as compare to previous generation.
e) Fitness / generation graph was not converging towards an optimal fitness value.

Experiment 1 was based on the decision of the user for modelling the branching structures. It was observed that there were several assumptions and observations taken by different users to terminate the procedure. For example:

a) According to the user, the distribution of gene values was equally distributed.
b) Most of the genes values' colour intensity was high (Dark Blue Colour), meaning that the majority of the search space had fitter solutions.
c) There was no any difference found between the distributions of gene values even with interactions.

d) There was not any difference found between the generated structures in the last few generations.

The colour intensity also played an important role for the decision of the interactions. The user observed the variation of colour and size of the ellipse with more interest as compared to the fitness convergence.

Analysis of the proposed approach for parameters depends on the frequency of genes at each gene location. In VIGA-2D during the search process, proposing new gene values at different locations helped to fill up the gap between the evolution processes. It was also noticed that the worse gene values at some locations were bringing the GA quite far from its target solutions. To overcome this problem, the interaction may have been an immediate action to bring the gene values closer to the better frequency in the next generations.

The impact of variation in population size was also analyzed. It was observed that the increment in population size also impacts on the performance of the searching process. It should be noted here that increment in population size depends on successful interactions by users. With each interaction, the search space becomes wider. In most of the experiments results it was noted that the resultant population size was 50% larger than the initial size. Hence in VIGA-2D, dynamic population size was based on using wider search space interactively.

On the other hand, the proposed methodology gives a chance for GA to make a decision to absorb the proposed values as a part of the evolution or to discard it. In this way, the user performance could easily be monitored with the effect on population size. It was also observed that proposing less optimal gene values or allocating null values, directly impacts on the performance of the algorithm, i.e. it may produce unnecessary noise and distortion in search space. But due to the usage of the fitness function for assigning fitness, the GA may solve this problem in few generations or more quickly by inviting more user interactions.

4.8 Summary of Chapter

The proposed approach has been evaluated with both objective and subjective experiments. For benchmark functions, the proposed approach has been compared with SGA with having two different variables lengths and with 10 runs for each chromosome length. Hence, 10 runs for each experimental result with two different variable lengths ensure that the proposed approach is thoroughly evaluated. The presented results for benchmark functions were based on average result after 10 runs. The results show that the proposed approach outperformed SGA especially in large variable lengths. Through different experiments and observations, it was found that the 2-D graph visualization technique based on gene distributions is a better technique to monitor all changes going on in the search space. By using 2-D visualization technique, interaction of user, which is, based on gene values can be controlled and monitored easily and clearly. The user can monitored the gene values at different locations using 2-D Graph. Further, a 2-D graph also helps to look into details the process of the searching behaviour used by GA, specifically, which part of search space has optimal or less optimal gene values.

For testing the performance of VIGA-2D with parameters, different variable lengths of chromosomes were input to VIGA-2D. Different selection methods, crossover and mutation rates were also used to do the experiments. Multiple runs of VIGA-2D also show that there is a difference in the performance of GA by using different operators or their values.

The evaluation of results with different problems and inputs shows that this approach successfully converges to an optimum / fitter solution. It is also observed in the analysis, that the proposed approach able to display the complete picture of the multidimensional search space of GA. Hence, the visualization of gene values on 2-D graph would be a good technique to show the frequency of genes at any particular location. The next chapter concludes this thesis by summarizing the work and describing the main contributions.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Chapter Overview

This chapter concludes this research by presenting a summary of the work presented in this thesis. The strength of the proposed approach with different experimental results is also briefly discussed. Some suggestions about possible areas of future research directions are briefly outlined in this chapter.

5.2 Thesis Summary

This thesis has described a novel approach for the visualization of multidimensional GA data to 2-D space. This visualization is based on the displaying of gene values of each generation on a 2-D graph.

In the proposed research work, for accelerating the performance of the GA an idea of proposing a new individual into the search space is introduced. This new proposed individual then becomes a part of the next generation after an evaluation test. This evaluation test is for analysing the fitness of the proposed individual so that it will take an active part in the next generation. Since, for existing IGA techniques, the main problem was to reduce the user's fatigue [15]; so, in proposed approach, the human interaction is not forced in every generation, and fitness is calculated using the fitness function. A dynamic population size is used to make the proposed individual part of the search space in the next generation. The termination criterion of the proposed approach is on user perception or on generation number for an optimal solution. The graphical interface is designed in such a way that it gives the complete knowledge of converges from generation to generation. An interactive tool named VIGA-2D has been developed for this purpose.

5.2.1 Visualization of Genetic Algorithm on 2-D Graph

VIGA-2D is an interactive tool developed for visualization of a multidimensional data on a 2-D graph. The vertical view of this graph represents the gene values and the horizontal view represents the gene locations. The user directly interacts on the graph for proposing new values.

The graphical interface is designed in such a way that it gives the complete knowledge of converges from generation to generation. For example the gene values visualize with different size and different depth for blue colour according to their fitter status in search space. In which, gene values having high fitness value are displayed with large size and high depth colour for blue as compare to gene values with worse fitness displayed with small size and having light depth colour for blue (gray). Beside this, fitness versus generation graph is visualized for understanding and analyzing convergence of GA. This graph shows the visualization of the worst, average and best fitness in every generation.

5.2.2 Evaluation of Experimental Results

The performance of VIGA-2D had been analyzed with the help of subjective and objective analysis. For objective analysis, the experimental results were based on 2 benchmarks functions. The performance evaluation of VIGA-2D was done with the comparison of SGA using the same parameters. Secondly, the VIGA-2D was evaluated by 5 users participating in the experiment in order to visualize the test for subjective analysis. For this purpose, real values were evolved by the proposed approach to derive the rules for the Parametric L-System. The experiments performed with VIGA-2D shows that it has the ability to perform efficiently as compared to SGA. However it is also concluded that the performance of VIGA-2D depends on the optimal user's interactions. In other words, user's participation in several generations in the form of proposing a new individual helps to accelerate the performance of GA.

5.3 Future Work

Working on the completion of this thesis has generated several interesting and promising ideas which will be explored in the future to address the problem of visualization of the Interactive Genetic Algorithm more efficiently. Moreover, the future work described in this section will further be investigated for better understanding of the visualization technique for GA.

A key area where VIGA-2D can be improved is the way it helps a user to do interactions. At present, VIGA-2D concludes that the user does the interactions based on understanding the difference of size and colour of gene values in current generation. This approach may be improved by drawing a line between gene values having same fitness value. This line may help to show the relationship between gene values located in the same individual.

A way forward would be to use clustering technique to map gene values according to their fitness on 2-D graph. The user then could interact on each cluster to propose a new gene value. Hence proposed gene values will work as an individual in the next generation.

A further enhancement would be implementing a sketch interface for generating initial branching structure L-System. During subjective analysis of the proposed approach it was observed that the users found difficulty to input L-string and initial parameters. However, besides this manual input a user interface may be implemented to sketch the initial structure to be further evolved by VIGA-2D. In this way, the user should not need to learn the complex grammar of the L-system to run VIGA-2D.

LIST OF REFERENCES

[1] H. C. Purchase, *et al.*, "Theoretical Foundations of Information Visualization," *Information Visualization: Human-Centered Issues and Perspectives*, Springer-Verlag, pp. 46-64, 2008.

[2] C. D. Hansen and C. R. Johnson, *The visualization handbook*: Academic Press, 2005.

[3] H. Wright, *Introduction to Scientific Visualization*: Springer-Verlag New York, Inc., 2006.

[4] D. A. Keim, "Visual exploration of large data sets," *Commun. ACM,* vol. 44, pp. 38-44, 2001.

[5] J. H. Holland, "Adaptation in natural and artificialsystems, *"MITPress*, 1992.

[6] A. Eiben, *et al.*, "Global convergence of genetic algorithms: A Markov chain analysis," *Parallel Problem Solving from Nature,* pp. 3-12, 1991.

[7] Z. Michalewicz, *"Genetic algorithms + data structures = evolution programs,"Springer-Verlag New York, Inc.*, 1994.

[8] C. R. Darwin, "On the Origin of Species By Means of Natural Selection,". *London*, 1859.

[9] D. E. Goldberg, "Genetic Algorithms in Search, Optimization and Machine Learning," *Addison-Wesley Longman Publishing Co., Inc.,* 1989.

[10] H. Nishino, *et al.*, "A 3D Modeler for Aiding Creative Work Using Interactive Evolutionary Computation," *IEICE Transactions on Information and Systems,* vol. 85, pp. 1473-1483, 2002.

[11] P. Haroun,"Genetic Algorithm and Data Visualization," MS thesis, School of Computer Science, McGill University, 1997.

[12] T. D. Collins, "Applying software visualization technology to support the use of evolutionary algorithms," *Journal of Visual Languages & Computing,* vol. 14, pp. 123-150, 2003.

[13] H. Pohlheim, "Visualization of Evolutionary Algorithms -Set of Standard Techniques and Multidimensional Visualization," *Proc Genetic and Evolutionary Computation Conference*, San Francisco, 1999, pp. 533-540.

[14] S.-B. Cho, "Towards Creative Evolutionary Systems with Interactive Genetic Algorithm," *Applied Intelligence,* vol. 16, pp. 129-138, 2002.

[15] N. Hayashida and H. Takagi, "Acceleration of EC convergence with landscape visualization and human intervention," *Applied Soft Computing,* vol. 1, pp. 245-256, 2002.

[16] N. Hiroaki, *et al.*, "An Interactive 3D Graphics Modeler Based on Simulated Human Immune System," *Journal of Multimedia,* vol. 3, pp. 51-60, 2008.

[17] Z. Gu, *et al.*, "Capturing aesthetic intention during interactive evolution," *Computer-Aided Design,* vol. 38, pp. 224-237, 2006.

[18] M. Mach and Z. Zetakova, "Visualising Genetic Algorithms: A Way through the Labyrinth of Search Space," *Intelligent technologies: theory and applications: new trends in intelligent technologies,* pp. 279, 2002.

[19] S.-I. Ito, *et* al., "A Visualization of Genetic Algorithm Using the Pseudo-color," *Neural Information Processing:14th International Conf Kitakyushu, Japan,Springer-Verlag,* 2008, pp. 444-452.

[20] T. Routen, "Techniques for the visualisation of genetic algorithms," *IEEE Conference on Evolutionary Computation.*, Piscataway, New Jersey, 1993, pp. 846-851.

[21] H.-J. Min and S.-B. Cho, "Creative 3D Designs Using Interactive Genetic Algorithm with Structured Directed Graph,", 2004, pp. 391-400.

[22] K. Aoki and H. Takagi, "3-D CG lighting with an interactive GA," *First Int Conf on Knowledge-Based Intelligent Electronic Systems,1997,* pp. 296-301.

[23] M. S. Samhouri, "An intelligent opportunistic maintenance (OM) system: a genetic algorithm approach," 2010, pp. 60-65.

[24] O. Bandte, "A broad and narrow approach to interactive evolutionary design--An aircraft design example," *Applied Soft Computing,* vol. 9, pp. 448-455, 2009.

[25] H. Takagi and K. Ohya, "Discrete fitness values for improving the human interface in an interactive GA," *Proc IEEE International Conference Evolutionary Computation,*1996, pp. 109-112.

[26] D. Gong, *et al.*, "Intercative Genetic Algorithm with Individual Fitness not assigned by Human," *Journal of Universal Computer Science,* vol. 15, pp. 2446--2462, 2009.

[27] D. R, "The evolution of evolvability,artificial life," in *Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems,* Redwood City,CA, 1989.

[28] H. Takagi, "Interactive evolutionary computation: fusion of the capabilities of EC optimization and human evaluation.,"*Proc of the IEEE,* 2001,pp. 1275-1296.

[29] C. Clare B and R. H. Mazza, "GenTree: An Interactive Genetic Algorithms System for Designing 3D Polygonal Tree Models," in *Genetic and Evolutionary Computation,* 2003,p. 216.

[30] H. Nishino, *et al.*, "A 3D modeling system for creative design," *15th Intl Conf Information Networking,* 2001, pp. 479-486.

[31] A. M. Brintrup, *et al.*, "An interactive genetic algorithm-based framework for handling qualitative criteria in design optimization," *Computers in Industry,* vol. 58, pp. 279-291, 2007.

[32] H. Nishino, *et al.*, "A Ubiquitous 3D Graphics Modeler for Mobile Devices," in *International Symposium on Parallel and Distributed Processing with Applications*, 2008.

[33] R. Curry, "On the Evolution of Parametric L-systems," 2000.

[34] M. Bruce, *et al.*, "Genetic selection of parametric scenes," University of Cape Town, 2003.

[35] Seung-Hwan Choi, *et al.*, "Interactive Genetic algorithm for designing the appearance of software robot using homologous chromosome representation,"*17th World Congress of Automatic Control*, Seoul, Korea, 2008.

[36] J. Graf and W. Banzhaf, "Interactive Evolution for Simulated Natural Evolution," *European conf on Artificial Evolution*, 1996.

[37] A. Oliver, *et al.*, "Interactive Design of Web Sites with a Genetic Algorithm," in *ADIS Int Conf WWW/Internet*, 2002,pp. 355-362.

[38] J. Ren, *et al.*, "Interactive genetic algorithms with variational population size," *Emerging Intelligent Computing Technology and Applications with Aspects of Artificial Intelligence,* pp. 64-73, 2009.

[39] H.-S. Kim and S.-B. Cho, "Application of interactive genetic algorithm to fashion design," *Engineering Applications of Artificial Intelligence,* vol. 13, pp. 635-644, 2000.

[40] H. Nishino, *et al.*, "A virtual modeling system for intuitive 3D shape conceptualization," *IEEE Int Conf onSystems, Man and Cybernetics*, 2002, pp. 6.

[41] R. Jaksa and H. Takagi, "Tuning of image parameters by interactive evolutionary computation," *IEEE Int Conf on Systems, Man and Cybernetics*, 2003, pp. 492-497.

[42] H. Takagi, "Interactive Evolutionary Computation as Humanized Computational Intelligence Technology," *Proc Conf, 7th Fuzzy Days on Computational Intelligence, Theory and Applications*, 2001.

[43] G- Yi-nan at al., " Interactive genetic algorithms based on frequentpattern mining ," *Sixth International conf on Natural Computation (ICNC)*, 2010, pp. 2381 – 2385.

[44] John A. Biles , *et al.*, "Neural Network Fitness Functions for a Musical IGA " in *Intl Symp onIntelligent Industrial Automation and Soft Computing*, 1996, pp. 39-44.

[45] M. Ohsaki and H. Takagi, "Improvement of presenting interface by predicting the evaluation order to reduce the burden of human interactive EC operators,"*IEEE Int Conf on Systems, Man, and Cybernetics*, 1998, pp. 1284-1289.

[46] D. E. Goldberg, "Sizing Populations for Serial and Parallel Genetic Algorithms," *Proc of the 3rd Int Conf on Genetic Algorithms*, 1989.

[47] J. Arabas, *et al.*, "GAVaPS-a genetic algorithm with varying population size," *Proc IEEE Conf on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, 1994,pp. 73-78.

[48] Eiben, AE,*et al.*, "Evolutionary Algorithms with on-the-fly Population Size Adjustment." , *Springer,* vol. 3242, 2004, pp. 41-50.

[49] G. Kokai, *et al.*, "Parametric L-system description of the retina with combined evolutionary operators," in *Genetic and Evolutionary Computation Conf, GECCO*, Orlando, Florida,USA., 1999.

[50] M. Mitchell, "*An Introduction to Genetic Algorithms."MITPress*, 1996.

[51] L. Jong-Ha, *et al.*, "Accelerating evolution by direct manipulation for interactive fashion design," in *Proceedings. of Fourth International Conference on Computational Intelligence and Multimedia Applications*, pp. 362-366, 2001.

[52] T. Mutoh, *et al.*, "An experimental study for automatically generating image filter sequence by using simulated breeding,"Workshop on Interactive Evolutionary Computation, Fukuoka, Japan, 1998.

[53] K. Otoba, *et al.*, "Image prccessing and interactive selection with Java based on genetic algorithms,"*Workshop on Artificial Intelligence in Agriculture*, Makuhari, Japan, 1998, pp. 83-88.

[54] A. UUR, "Path planning on a cuboid using genetic algorithms," *Inf. Sci.,* vol. 178, pp. 3275-3287, 2008.

[55] J. B. S. Baik*et al.*, "Visualization for Genetic Evolution of Target Movement in Battle Fields," in *Computational Science – ICCS 2005*, *SpringerBerlin*,pp. 1064-1067.

[56] K. Onishi, *et al.*, "Interactive modeling of trees by using growth simulation," Proc ACM symp on Virtual reality software and technology, Osaka, Japan, 2003.

[57] D. J. Denis and M. Friendly, "*Milestones in the history of thematic cartography, statistical graphics, and data visualization*," 2006.

[58] K. C. Stuart, *et al.*, Eds., Readings in information visualization: using vision to think. *Morgan Kaufmann Publishers Inc.*, 1999

[59] J. J. Thomas and K. A. Cook, *Illuminating the Path: The Research and Development Agenda for Visual Analytics*: National Visualization and Analytic Centers, 2005.

[60] D. A. Keim, "Information Visualization and Visual Data Mining," *IEEE Transactions on Visualization and Computer Graphics,* vol. 8, pp. 1-8, 2002.

[61] F. Usama, *et al.*, "Information visualization in data mining and knowledge discovery,"*Morgan Kaufmann Publishers Inc.*, 2002.

[62] R. M. Rohrer and E. Swing, "Web-Based Information Visualization," *IEEE Comput. Graph. Appl.*, vol. 17, pp. 52-59, 1997.

[63] O. K. Osahon, "The Effect of Information Visualization on Financial Reports," Masters Thesis, School of Economics, HANKEN, 2008.

[64] Gillilan, "Scientific visualization of chemical systems," 1993, pp. 296-301.

[65] R. B. Haber, *et al.*, "A data model for scientific visualization with provisions for regular and irregular grids," *Proc of the 2nd conf on Visualization*, San Diego, California, 1991.

[66] G M Lingaraju, *et al.*, "Real Time Scientific Visualization of the ALE Based Free Surface Simulation for Virtual Reality Applications," *IJCSNS Int Journal of Computer Science and Network Security,* vol. 8, pp. 399-402, 2008.

[67] W. Ovtscharoff, *et al.*, "3D-Brain Model Software A New Interactive Real-Time Graphics Visualization for Rat Brain," 2009.

[68] M. Elke, "Interactive 3D Protein Structure Visualization Using Virtual Reality," 2004, pp. 503-503.

[69] S. Zhang, *et al.*, "An immersive virtual environment for DT-MRI volume visualization applications: a case study," *Proc of the conf on Visualization*, San Diego, California, 2001.

[70] T. D. Collins, "Using Software Visualization Technology to Help Evolutionary Algorithm Users Validate their Solutions.,"*7th IntConf on Genetic Algorithms*, Michigan, 1997, pp. 307-314.

[71] Y.-H. Kim and B.-R. Moon, "New usage of Sammon's mapping for genetic visualization," Proc of the Int Conf on Genetic and evolutionary computation: PartI, Chicago, IL, USA, 2003.

[72] H. Pohlheim, "Multidimensional Scaling for Evolutionary Algorithms-Visualization of the Path through Search Space and Solution Space Using Sammon Mapping," *Artificial Life*, vol. 12, pp. 203-209, 2006.

[73] W. B. Shine and C. F. Eick, "Visualizing the evolution of genetic algorithm search processes," *IEEE Int Conf onEvolutionary Computation,*1997, pp. 367-372, 1997.

[74] T. D. Collins, "Understanding evolutionary computing: a hands on approach," *IEEE Int Conf on Evolutionary Computation, IEEE World Congress on Computational Intelligence.*, 1998, pp. 564-569.

[75] D. F. Swayne, *et al.*, "Interactive Dynamic Data Visualization in the X Window System.," *Journal of Computational and Graphical Statistics,* vol. 7, pp. 113--130, 1998.

[76] T. Routen, "Techniques for the visualisation of genetic algorithms," in *IEEE World Congress on Computational Intelligence., Proc First IEEE Conf on Evolutionary Computation*, 1994 pp. 846-851.

[77] O. Deussen and B. Lintermann, *"Digital Design of Nature: Computer Generated Plants and Organics,"* SpringerVerlag, 2004.

[78] A. Lindenmayer, "Mathematical models for cellular interactions in development.I. Filaments with one-sided inputs," *Journal of Theoretical Biology,* vol. 18, pp. 280-299, 1968.

[79] H. Honda, "Description of the form of trees by the parameters of the tree-like body: Effects of the branching angle and the branch length on the shape of the tree-like body," *Journal of Theoretical Biology,* vol. 31, pp. 331-338, 1971.

[80] P. Prusinkiewicz and A. Lindenmayer, "The algorithmic beauty of plants,"*Springer-Verlag,*1990.

[81] S. Manousakis, "Musical L-Systems " Master's Thesis, The Royal Conservatory, The Hague, 2006.

[82] J. Hanan, "Parametric L-systems and Their Application To the Modelling and Visualization of Plants," Ph.D. dissertation, University of Regina, 1992.

[83] M. Grubert. Simulating plant growth *Crossroads The ACM Student Magazine*.

[84] B. Runqiang, *et al.*, "Derivation of L-system Models from Measurements of Biological Branching Structures Using Genetic Algorithms," Proc 15th Int conf on Industrial and engineering applications of artificial intelligence and expert systems: developments in applied artificial intelligence, 2002.

[85] H. Noser, *et al,*."Rule-Based Animation System With Genetic Algorithms As Test-Bed For Generic Evolutionary Applications",2002.

[86] N. Zakaria, "A Sketch-and-Grow Interface for Botanical Tree Modeling," *SpringerVerlag*, 2010, pp. 25-32.

[87] G. Kokai, *et al.*, "Evolving artificial trees described by parametric L-systems," in *IEEE Canadian Conference on Electrical and Computer Engineering*,1999, pp. 1722-1727.

[88] G. Ochoa, "On Genetic Algorithm and Lindenmayer System," *ProcInt Conf on Parallel Problem Solving from Nature*, 1998, pp. 335-344.

[89] Y. Rodkaew, *et al.*, "Modeling Leaf Shapes Using L-systems and Genetic Algorithms," *International Conference NICOGRAPH*,2002, pp. 73-78.

[90] L E. Da Costa at al., "Generating grammatical plant models with genetic algorithms," *Proc of ICANNGA*, Published by Springer,2005, pp 230-235.

[91] A. Daniel et al, "Simultaneous Evolution of Bracketed L-system Rules and Interpretation," *IEEE Congress on Evolutionary Computation*, Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada, 2006.

[92] L. Yang, "Aesthetic Evolution of Staged L-systems for Tiling Pattern Design," *Int Sym on Intelligent Information Technology and Security Informatics*, 2010, pp. 488-492.

[93] C.-Y. Liou, *et al.*, "Modeling complexity in musical rhythm," *Complexity,* vol. 15, pp. 19-30, 2010.

[94] T. Ijiri, *et al.*, "The sketch L-system: global control of tree modeling using free-form strokes,"2006, pp. 138-146.

[95] F. Anastacio, *et al.*, "Modeling plant structures using concept sketches," *ACM,* 2006, p. 113.

[96] G. S. Hornby and J. B. Pollack, "Evolving L-systems to generate virtual creatures," *Computers & Graphics,* vol. 25, pp. 1041-1048, 2001.

[97] A. Gülbeden, "L-Systems for Plant Generation," *BilkentUniversity*, 2001.

[98] L. Chen, *et al.*, "Adaptive L-system," (Nove 10, 2008 [Online].

Available: http://old.mee.chu.edu.tw/labweb/CGW2003/Session/b5/b5_2.pdf

[99] G. K, *et al.*, "Modelling Blood Vessels of the Eye with Parametric L-Systems Using Evolutionary Algorithms," *Proc the Joint European Conference on Artificial Intelligence in Medicine and Medical Decision Making*, 1999.

[100] R. Durikovic, *et al.*, "Animation of Biological Organ Growth Based on L-systems," *Computer Graphics Forum,* vol. 17, pp. 1-13, 1998.

[101] (March 08-2009). *CMIVFX Houdini L-Systems Essentials 1,2.* Available: http://www.heroturko.org/3d/1693-cmivfx-houdini-l-systems-essentials-12.html

[102] Y. I. H. Parish, *et al.*, "Procedural modeling of cities," *Proc of the 28th annual conference on Computer graphics and interactive techniques*, 2001.

[103] J. McCormack, "Interactive evolution of L-system grammars for computer graphics modelling," *Complex Systems: from biology to computation,* pp. 118-130, 1993.

[104] P. Prusinkiewicz, *et al.*, "The use of positional information in the modeling of plants,"*Proc of the 28th annual conference on Computer graphics and interactive techniques,* 2001, pp. 289-300.

[105] T. Hu. (June 26, 2010),"*Variable Population Size and Evolution Acceleration: A Case Study with A Parallel Evolutionary Algorithm*". Available: www.evolutioninmaterio.com/preprints/vps.pdf

[106] D. E. Goldberg, *et al.*, "Genetic Algorithms, Noise, and the Sizing of Populations," *COMPLEX SYSTEMS,* vol. 6, pp. 333-362, 1991.

[107] S. Legg, *et al.*, "Tournament versus fitness uniform selection," *Arxiv preprint cs/0403038,* 2004.

[108] L. D. Davisl, (1991). "Handbook of Genetic Algorithms," Available : Amazon.net.

[109] A. S. Wu, *et al.*, "Empirical observations on the roles of crossover and mutation," *Ann Arbor,* vol. 1001, p. 48109.

[110] K. D. Herbert Dawid et al.,"Quantitative models of learning organizations",1st ed.: Springer, 2002.

[111] P. Prusinkiewicz, "Graphical applications of L-systems,"Proc on Graphics Interface '86/Vision Interface '86, Vancouver, British Columbia, Canada.

[112] P. Prusinkiewicz, "Applications of L-systems to computer imagery,"*Proc 3rd Int Workshop on Graph-Grammars and Their Application to Computer Science*, 1987.

[113] K. D. D. Jong, "An analysis of the behavior of a class of genetic adaptive systems," PhD, Departament of Computer and Communication Sciences, Ann Arbor, University of Michigan, 1975.

[114] H. H. Rosenbrock, "An automatic method for finding the greatest or least value of a function," *Computer Journal,* vol. 3, pp. 175-184, 1960.

[115] D. B. Fogel, "Evolutionary computation: toward a new philosophy of machine intelligence,"*Wiley-IEEE Press*, 2006.

[116] J. David Schaffer et al., "A study of control parameters affecting online performance of genetic algorithms for function optimization," Proc of the 3rd International Conference on Genetic Algorithm, San Francisco, CA, USA, 1989.

[117] T. Back, Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms: The Clarendon Press Oxford University Press, 1996.

[118] O Gabriela et al., "Optimal Mutation Rates and Selection Pressure in Genetic Algorithms," Proc. Genetic and Evolutionary Computation Conference (GECCO), 2000

[119] P. Hogeweg and B. Hesper, "A model study on biomorphological description," *PatternRecognition*, vol. 6, pp. 165-179, 1974.

[120] D. Frijters and A. Lindenmayer, "A Model for the Growth and Flowering of Aster Novae-Angliae on the Basis of Table < 1, 0 > L-Systems," conf in Aarhus, Denmark, 1974.

[121] A. R. Smith, "Plants, fractals, and formal languages," *SIGGRAPHComput. Graph.*, vol. 18, pp. 1-10, 1984.

[122] S. Papert, "Mindstorms: Children, computers and powerful ideas," New York, 1980.

[123] J. S. Hanan, "PLANTWORKS: A software system for realistic plant modelling," Master's thesis, University of Regina, 1988.

[124] J. D. Foley and A. V. Dam, Fundamentals of interactive computer graphics: Addison-Wesley Longman Publishing Co., Inc., 1982.

LIST OF PUBLICATIONS

Humera Farooq, Dr M Nordin Zakaria,Dr Mohd Fadzil Hassan, Dr Suziah Sulaiman, "An Interactive Visualization of Genetic Algorithm on 2-D Graph", The 10th IEEE International Conference on COGNITIVE INFORMATICS & COGNITIVE COMPUTING (ICCI*CC'11), August 18-20, 2011, Banff, Alberta, Canada.

Humera Farooq, Dr M Nordin Zakaria,Dr Mohd Fadzil Hassan, Dr Suziah Sulaiman, "Representation of Multidimensional data using Interactive Genetic Algorithm", International Symposium on Information Technology (ITSim'10), June 15-17, 2010, Kuala Lumpur, Malaysia.

Humera Farooq, Dr M Nordin Zakaria, Dr Mohd Fadzil Hassan, Dr Suziah Sulaiman, "An Approach to Derive Parametric L-System using Genetic Algorithm", 1st International Visual Informatics Conference 2009 (IVIC) 12th – 13th November 2009, Kula Lumpur , Malaysia, LNCS, Volume 5857, pp. 455-466, 2009.

Humera Farooq, Dr M Nordin Zakaria, Dr M Fadzil Hassan, Dr Suziah Sulaiman, "Hierarchical Genetic Algorithms for the Derivation of L-Systems", 4th International Conference On Information Technology & Multimedia 2008 (ICIMu) 18th – 19th November 2008, Kula Lumpur, Malaysia.

APPENDIX A

DESIGN AND IMPLEMENTATION

A friendly GUI makes the usage of GA more interesting. This appendix discusses all the important aspects and features of the design tool named VIGA 2-D. This tool is implemented using Java programming with 2 and 3D Graphics.

A.1 INTERACTIVE VISUALIZATION TOOL (VIGA-2D)

VIGA-2D is based on the GA methods and operators described in Chapter 3. When VIGA-2D starts, a window for setting up all the parameters and methods is loaded onto the screen. Figure A.1 shows all the important components and data flow between them. The details for all the important functions are described in the following sections.

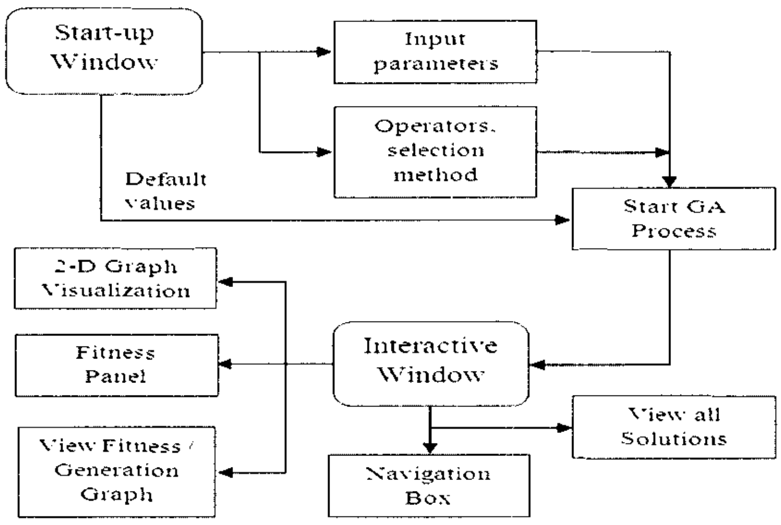
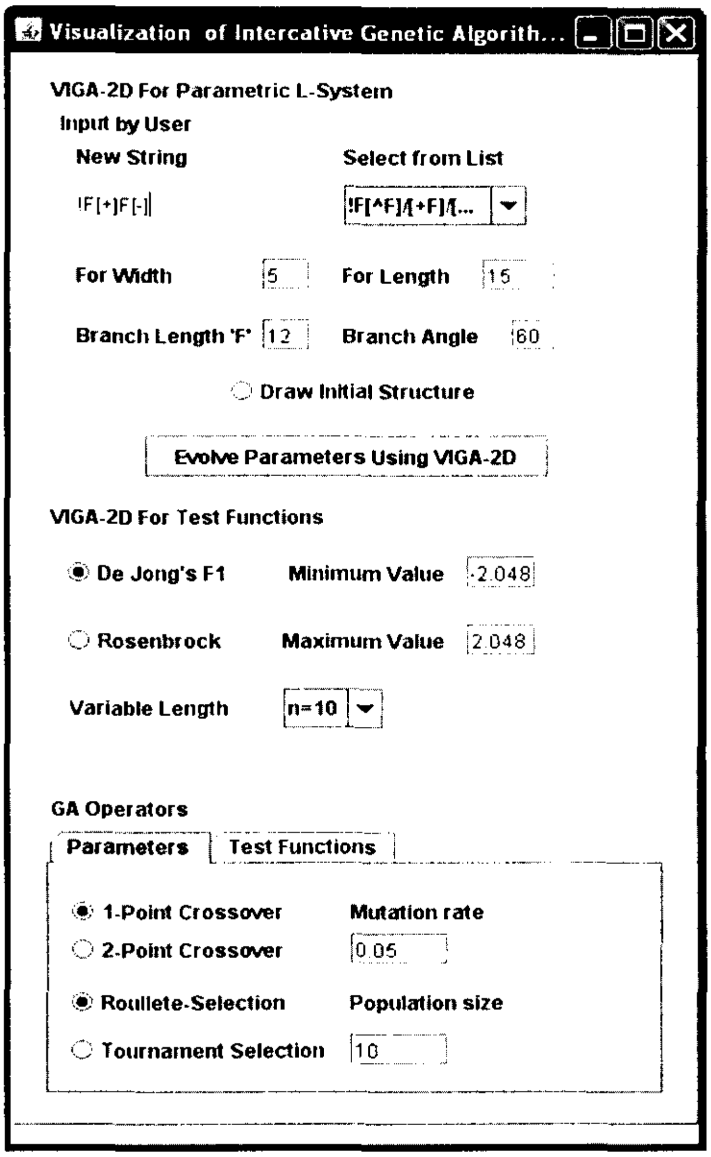


Figure A.1. GUI Design for Proposed Approach


Visualization of Intercative Genetic Algorith...

VIGA-2D For Parametric L-System

Input by User

New String: !F[+]F[-]

Select from List: !F[^F]/[+F]/[...

For Width: 5 For Length: 15

Branch Length 'F': 12 Branch Angle: 60

Draw Initial Structure

Evolve Parameters Using VIGA-2D

VIGA-2D For Test Functions

[✓ De Jong's F1] Minimum Value: -2.048

Rosenbrock Maximum Value: 2.048

Variable Length: n=10

GA Operators

Parameters | Test Functions

[✓ 1-Point Crossover] Mutation rate: 0.05

2-Point Crossover

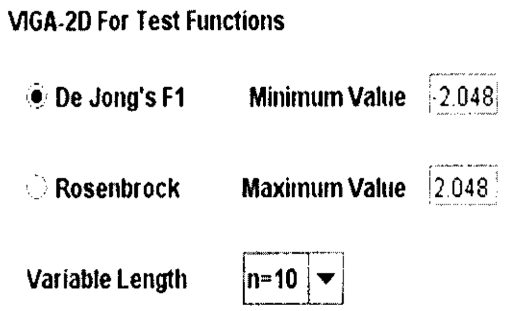
[✓ Roullete-Selection] Population size: 10

Tournament Selection

A.2 Start-up Window

The start-up window is the first window of the VIGA-2D tool as shown in Figure A.2. This window allows the user to give the input values or to select different GA operators and selection methods. By default values are given to initialize all parameters, GA operators and selection methods. If there is no input by the user than the GA process starts with these default values. Different dialog boxes are used to control the correct input and to check empty input fields.

VIGA-2D works for the L-System and tests functions with the same GUI. If the test functions tab is selected then the next selection is for choosing the function to be evolved by VIGA-2D. There are two different variable lengths, i.e. 10 and 20 for benchmark functions. Text fields are used to take input for the minimum and maximum values from the user. A GA parameters' panel is implemented to take inputs in all GA operators, to select the desired selection method and to assign the population size to initialize the process.



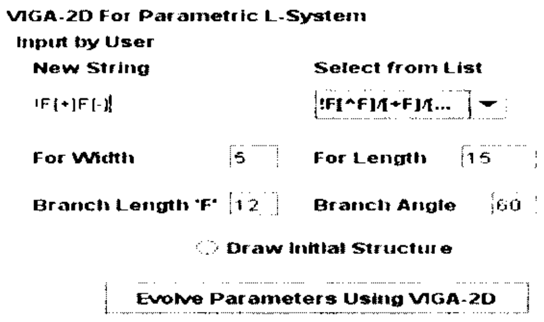
VIGA-2D For Test Functions

● De Jong's F1 Minimum Value -2.048

○ Rosenbrock Maximum Value 2.048

Variable Length n=10 ▼

If the Parametric L-System is selected then the initial parameters and string for generating the branching structure is input by the user or default values are used to initialize the GA process. A combo box is implemented proposing different strings for symbols with different lengths. The maximum length of the string allowed is 15.



VIGA-2D For Parametric L-System

Input by User

New String: !F[+]F[-]

Select from List: !F[^F]/[+F]/[... ▼

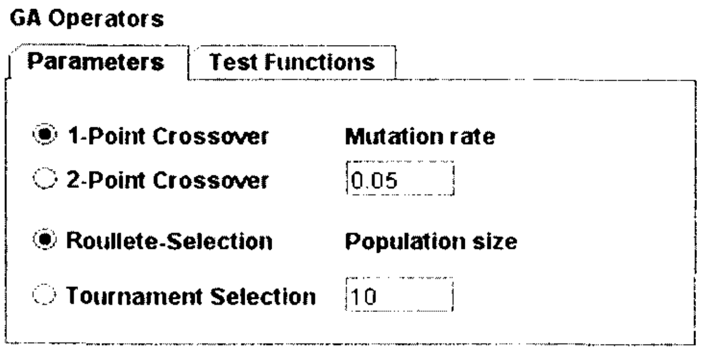
For Width 5 For Length 15

Branch Length 'F' 12 Branch Angle 60

○ Draw Initial Structure

Evolve Parameters Using VIGA-2D

For parameters, the overall value range is from 5.0 to 49.99. For GA parameters default values are given and can be changed by the user. There is an option to select the selection method which is the Roulette selection method and Tournament selection method.


GA Operators

Parameters | Test Functions

[✓ 1-Point Crossover] 2-Point Crossover

[✓ Roullete-Selection] Tournament Selection

Mutation rate: 0.05

Population size: 10

After input of all the parameters, the next process is to initialize the GA process. For test functions, the interactive window becomes enabled for the selected function. While for the Parametric L-System, the GA process is started with the randomly generated chromosome. The radio button "Draw Initial Structure" is used to see the branching structure generated by the user input.

A.3 Interactive Window

After selecting different options from the navigation window, an interactive window is loaded onto the screen. This interactive window consists of a navigation box, fitness panel, fitness / generation graph, text area with all individuals of the current generation and the interactive 2-D graph as shown in Figure A.3. The fitness panel, fitness/ generation graph and text area with all individuals is implemented only to view the progress and functionality of GA from generation to generation. The interactive 2-D graph can be interacted by the user. The navigation box has all the functionality to control the user interactions. Four different colour labels are used to show the intensity of the individuals in the search space. Lighter colours show the worst solution whereas dark colours show the best solution. Two labels are used to analyze the fitness of individuals in the current generations as compared to the previous generation. Overall, the performance and convergence of solutions from generation to generation depends on the user.

a. **Navigation Box**:

The navigation box shows the current generation number and population size. The next generation button is used to bring a new generation onto the screen, whereas the "Update" button is used to update the current generation with the proposed values. For the Parametric L-System, the navigation box has the "draw tree" button used to generate the branching structure with the best solution of the current generation.

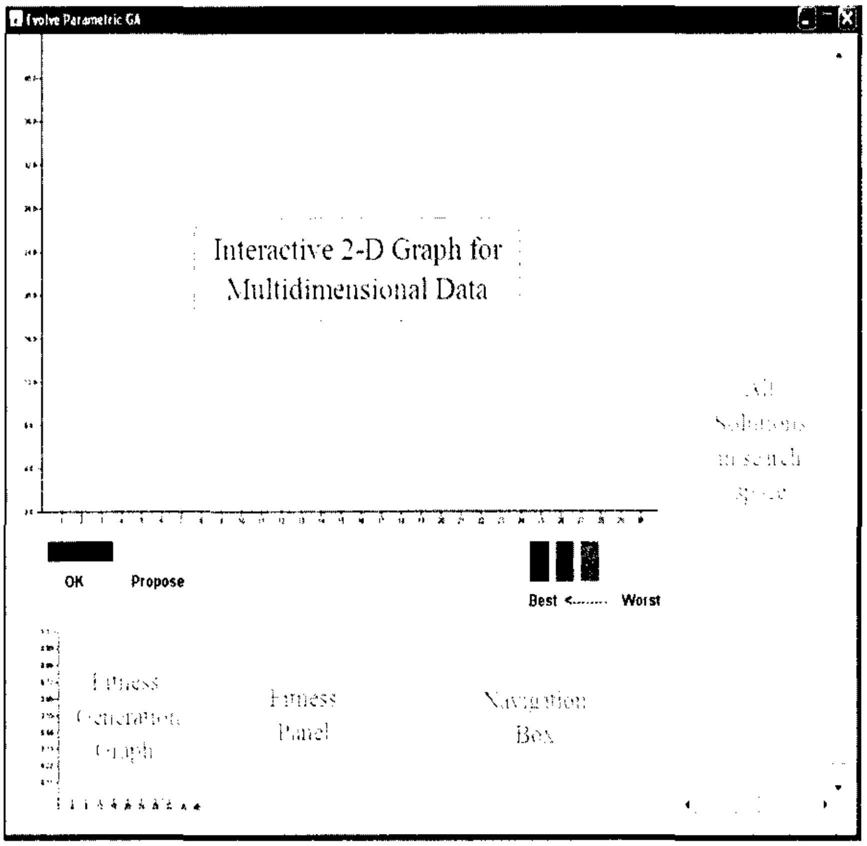
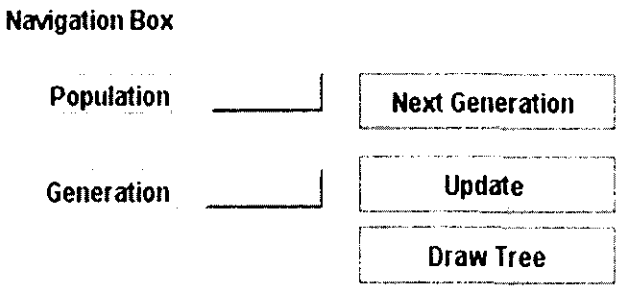
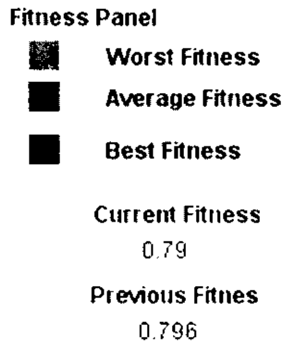


Figure A.3. Designed Interactive Window



b. Fitness Panel:

The purpose of implementing the fitness panel is to show the fitness of the best solution for the current generation and previous generation on the screen. In this way, it becomes easier for a user to understand the flow and the expected fitness values in the next generation. Hence, it is also helpful to show the convergence rate of GA towards an optimal solution. Three labels for best, average and worst fitness are also placed in this panel to monitor the fitness/generation graph accordingly.


Fitness Panel

■ Worst Fitness

■ Average Fitness

■ Best Fitness

Current Fitness

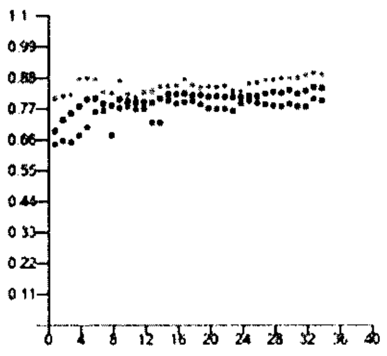
0.79

Previous Fitnes

0.796

c. Fitness/Generation Graph:

The fitness versus generation graph was implemented based on the worst, best and average fitness. This graph is also helpful to the user in understanding the GA convergence graphically. The upper and lower boundaries of the x-axis are computed according to the range of the value. For the y-axis the higher and lower boundaries are given by default.



d. All Individuals of the current search space:

A text area is used to display all individuals of the current search space. This test area holds all the history of the previous generations in the form of solutions and their fitness values.

e. 2-D Graph:

The interactive 2-D graph is based on the array values and total length of the array. In the 2-D graph these array values are represented on the x and y-axis. On the y-axis, the representation of this graph is based on the minimum and maximum values of the array. While the x-axis shows the location of each element in the array according to the array length.

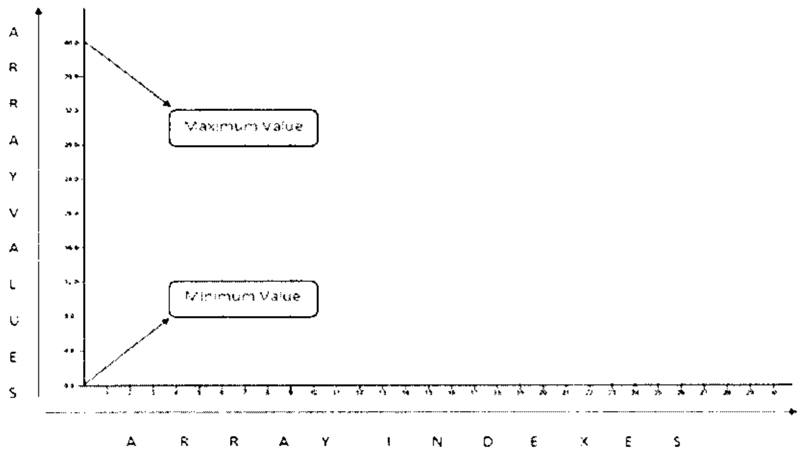


Figure A.4. 2-D Graph with X and Y-axis

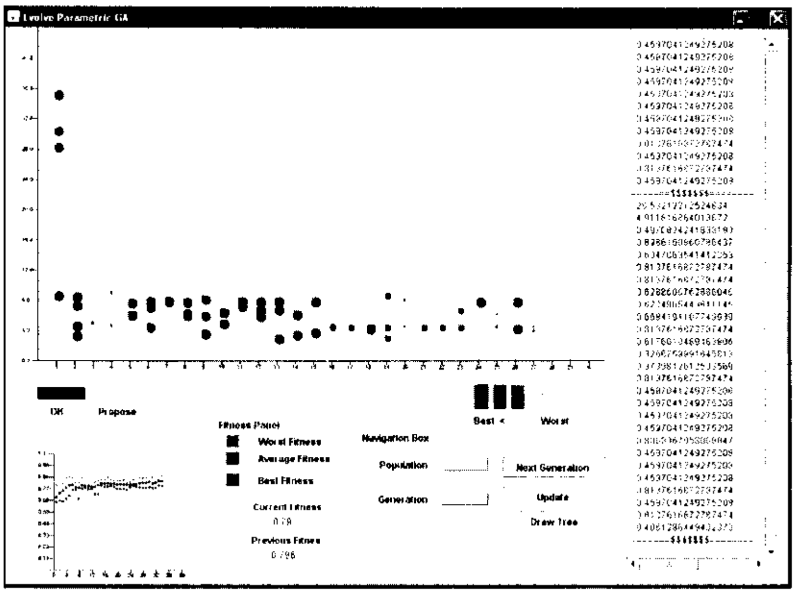
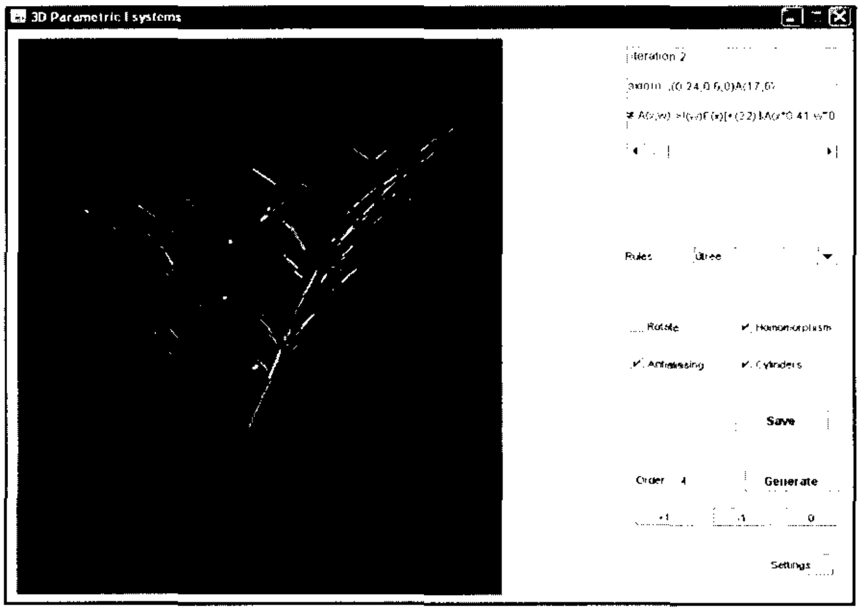


Figure A.5. Complete Interactive Window

A.4 Graphical window for Parametric L-System.

An interactive visualization 3-D space is implemented to draw the branching structures as shown in Figure A.6. For visualization of the output of GA, we have selected the best individuals from each generation. This selection is based on the higher fitness of individuals from a specific generation. One rule has been created from this best solution. The axiom is constant for all rules. The Red, Green, Blue (RGB) colour scheme is used in the axiom; each colour scheme is in the range of (0-1). Furthermore, the user can go to more or less iterations in the visualization window.

Light and shade effects are also given to create more natural looking structures. However, these parameters are constant and do not evolved using GA. The user can also rotate the generated structures in a 3-D view. Check box toggles can be activated by the user for rotation, to enable or disable the cylinder, for homomorphism and to apply ant aliasing. By default, the cylinder and homomorphism properties are enabled. 3 control buttons are used to increase, decrease or to initiate the iteration. The button having the '0' label shows the zero level of iteration which will return the axiom or it may call to reset all iterations. The setting button is used to change background colour, and enable or disable shadow and background.



Iteration 2

axiom ,(0.24,0.6,0)A(17,6)

A(x,w) >!(w)F(x)[+(22)]A(x*0.41 w*0

Rules Ltree

Rotate [✓ Homomorphism]

[✓ Antialiasing] [✓ Cylinders]

Order 4

+1 -1 0

Figure A.6. Graphical Window for the Parametric L-System

APPENDIX B

PARAMETRIC L-SYSTEM

The Parametric L-System works with the set of symbols, relational operators and asthmatic operations. However turtle geometry is used to interpret all these symbols' operators and parameters to construct the rules.

B.1 Parametric L-System

In the Parametric L-System, the parameters are associated with symbols. Let the alphabet be denoted by V, and the set of parameters is the set of real numbers R. A module with letter A $\in$ V and parameters (p1, p2 ....pn) $\in$ R is denoted by A (p1, p2 ....pn). Every module belongs to set Vx R*, where R* is a finite sequence of all parameters. The real-valued actual parameters appearing in words have a counterpart with formal parameters which may be used in the specification of L-system productions.

Let $\sum$ be the set of formal parameters. The combination of formal parameters and numeric constant using arithmetic operators (+, -, *, /), the logical operators (&&, ! , ||) , the relational operators (>, <, >=, <= , =) and parenthesis (()) will make a complete set of $\sum$ having all constructed logical and arithmetic expressions. These are noted as C ($\sum$) for logical expressions and E($\sum$) for arithmetic expressions. A Parametric OL-system is an ordered quadruplet G = (V,$\sum$, ω, P), where,

- *V* is the alphabet of the system.
- $\sum$ is the set of formal parameters.
- $\omega \in (V x R^*)+$ is a nonempty word called axioms.
- $P \; (V \; x \; \sum^*) \; x \; C \; (\sum) \; x \; (V \; x \; E(\sum)^*)^*$ is a finite set of productions.

In applying the production rules specified in the L-System, the predecessor will be recursively expanded and replaced with the relevant successors and it fulfils the condition if it is given by using arithmetic expressions. If there is no condition given, then it is considered as empty statement, hence, the resulting structure is drawn by using the deterministic Parametric L-System. The following example shows the deterministic Parametric L-System:

Axiom: $F(\alpha)$

Rule 01: $F \rightarrow F(\alpha_1)\,[-(\theta_1)\,F(\alpha_2)]\,F(\alpha_3)\,[+(\theta_2)\,F(\alpha_4)]\,\ldots\,F$

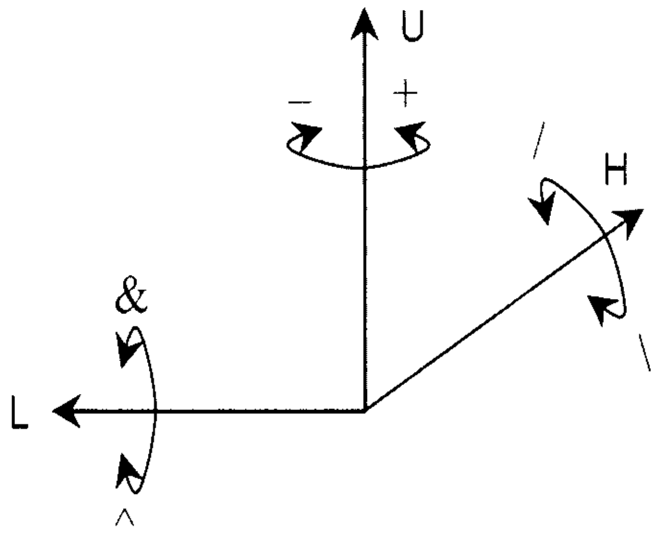
$(\alpha_n)\,[\,+(\theta_n)\,F(\alpha_n)]$

In the above example there is one module F in the axiom with the parameter α respectively. It consists of one production rule that rewrite the occurrence of F with the precise successor modules. Geometric Interpretations are used to manipulate these symbols to draw the structure onto the computer screen.

B.2 Geometric Interpretation

When using the L-System, two factors are very important to consider when generating any branching structure on the computer, i.e. (a) Development rules (control the growth) and (b) Geometric aspects of different angles [81]. Initially, the L-systems were conceived as a formal theory of development. Geometric aspects were not considered. Later, geometrical interpretations were proposed. Hogeweg and Hesper in 1974 [119] and Frijters and Lindenmayer in [120], for the first time discussed graphical interpretation for the L-System in their papers. Their work emphasised using the bracketed L-System as the branching topology for modelling plants. Geometric aspects were added in the post processing phase.

A later extension was found in the work of Smith [121] to use the L-System for modelling realistic images using computer graphic techniques. However, Prusinkiewicz for the first time added geometric commands directly with the L-System and extended these commands with the bracketed L-System [111] and in 2 or 3D space [112] . His work is based on LOGO turtle geometry [122] . A turtle is an object moved on a Graph (F). The current orientation of turtle in space is represented by 3 vectors: H, L, U, indicating the turtle's heading, the direction to the left, and the direction to the right. The symbols used to control this orientation are described in Table B.2. These turtle graphics are used to build a geometrical interpretation of L-system strings.



Turtle Interpretation in 3D space.

The discussion of turtle interpretation is derived from the work of Prusinkiewicz et a.l [80] and Hanan [123]. These turtle interpretations work either in 2 or 3-D space. All of these three orientations are perpendicular to each other, so by calculating any two of them, the third can be calculated by using the equation: L = H x U. Using this notation, the calculation for the rotation of the turtle is calculated using [H' L' U'] = [H L U] R.

Where, *R* is the 3 x 3 rotation matrix [124]. Standard rotation matrices are used to calculate the rotation in any direction around α angle.

$$R_H(\alpha \quad \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}$$

$$R_L(\alpha) = \begin{bmatrix} \cos\alpha & 0 & -\sin\alpha \\ 0 & 1 & 0 \\ \sin\alpha & 0 & \cos\alpha \end{bmatrix}$$

$$R_U(\alpha) = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Table B.1.Turtle Symbols used to draw in 3-D space.

Symbols	**Direction**	**Orientation**
F(s)	Move forward a step of length s	Draw a line
f(s)	Move forward a step of length s	Draw no line
+, -	Turn left and right	Matrix $R_U(\delta)$
&, ^	Pitch down and up	Matrix $R_L(\delta)$
\, /	Roll left and right	Matrix R_H
|	Turning around	Matrix $R_U(180^o)$
[,]	Push and pop the current state from stack.	Use for Branching

The input given by the user is restricted, controlled and formulated into the L-System language according to the symbols. The following are the symbols with their direction and orientation used to control the turtle position in the proposed application.

Moving and Drawing Symbols:

Symbol *'F'* is used to demonstrate the line to the L-System in the following manner:

- F(a) : Move forward a step of length a > 0. Draw a line from the previous status to the new one and the turtle position changes to (x', y', z'), where
  - $x' = x + \mathrm{a}H_x$,
  - $y' = y + \mathrm{a}H_y$,
  - $z' = z + \mathrm{a}H_z$.

Direction Commands:

The direction commands work when the user gives angles to either the left or right direction along with 90 degrees. However, the user may also give the direction commands for the z-axis. If the given branch angle is around the U vector and the given rotation angle $(\delta)$is smaller or larger than 90 then it calculates the following symbols and rotating angles:

- $+(\delta)$: If it rotates around U by an angle of δ degrees in the right direction.
- $-(\delta)$: If it rotates around U by an angle of δ degrees in the left direction.

If the given branch is around the L vector and the given rotation angle $(\delta)$is smaller or larger than 90 then it calculates the following symbols and rotating angles:

- $\&(\delta)$: Pitch down by an angle of δ degrees , Rotate around L .
- $\wedge\,(\delta)$: Pitch up by an angle of δ degrees, Rotate around L .

Commands for drawing attributes:

For drawing attribute, the thickness of the main trunk may also be given as a symbol and each branch also has a ratio for width.

- ! : Set the cylinder thickness value.

Branch Controlling Commands (Bracketed L-System):

- [: Push the current state into the stack
-] : Pop the current state from the stack.

APPENDIX C

INTERACTIONS FOR TEST FUNCTIONS

The tables in this appendix show the complete list of all interactions and total number of the interactions, and the average interaction for each run. Furthermore the proposed individuals accepted by the VIGA-2D and total number of the proposed individuals discarded by VIGA-2D for each test function are also listed here. It also shows the average number of discard and accepted proposed individuals in each run, where R1 to R10 represents each run.

Table C.1: Accepted and Discarded Interactions in all Runs for DeJongs' Function n=10

Interactions	Successful Interactions at Generation Number										Average Generation Number for Accepted Interactions
	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	
1	16	17	21	18	21	16	16	15	15	16	17.1
2	22		29	25	28	25	18	21	21	21	23.3
3	28	28	35	32	37	32	24	28	27	27	29.8
4	36	33	43	38	45	41	30	35	33	31	36.5
5	44	38	50	44	52		36	40	39	38	42.3
6	54	46		54		51	42	46	45	44	47.75
7		52	65		66	58	49	53		51	56.2
8		58		69		72	57	59	58		62.1
9			73	79	86			71		66	75
10	88		82	92	91			88	71	75	83.8
	Discarded Interactions at Generation Number										Average Generation Number for Discarded Interactions
1											0
2		25									25
3											0
4											0
5											0
6			61		60						60.5
7	60			62					52		58
8	67		68		73					59	66.7
9	76	71				86	66		64		72.6
10		76				89	71				78.6

Table C.2.Number of interactions, Successful Interactions, and Discarded Interactions for DeJong's Function n =10.

Parameters	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Average Generation Number for Interaction
Total Interactions	10	10	10	10	10	10	10	10	10	10	10
Accepted Interactions	7	7	8	9	8	7	8	10	8	9	8.1
Discard Interactions	3	3	2	1	2	3	2	0	2	1	1.9
Population Size after Interactions	17	17	18	19	18	17	18	20	18	19	18.1

Table C.3: Accepted and Discarded Interactions in all Runs for DeJong's Function n=20

Interactions	**Successful Interactions at Generation Number**										**Average Generation Number for Accepted Interactions**
	R1	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**	**R9**	**R10**	
1	17	14	17	15	16	17	17	19	18	18	16.8
2	23	20	24	23	21	25	26	26	26	22	23.6
3	28	28	31	32	26	33	40	32	34	29	31.3
4	32	37	38	38	35	39	50	40	42	34	38.5
5	37	46	45	46	43	47	61	48	54	42	46.9
6	43	58	51	53	51	51	72	58	63	54	55.4
7	50	68	55	62	57			67	74	60	61.6
8	55		66	69	61	65	87	76	85	69	63.3
9	66	87	74	80		71			94		78.6
10	88	92	88	88				91	96		90.5
	Discarded Interactions at Generation Number										**Average Generation Number for Discarded Interactions**
1											0
2											0
3											0
4											0
5											0
6											0
7						58	86				72
8		76									76
9					71		92	83		76	80.5
10					80	79	95			85	84.7

Table C.4. Number of Interactions, Successful Interactions, and Discarded Interactions for DeJong's Function n =20.

Parameters	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**	**R9**	**R10**	**Average Generation Number for Interaction**
Total Interactions	10	10	10	10	10	10	10	10	10	10	9.6
Accepted Interactions	10	9	10	10	8	8	7	9	10	8	8.9
Discarded Interactions	0	1	0	0	2	2	3	1	0	2	1.1
Population Size after Interactions	30	29	30	30	28	28	27	29	30	28	28.9

Table C.5: Accepted and Discarded Interactions in all Runs for Rosenbrock's Function n=10

Interactions	**Successful Interactions at Generation Number**										**Average Generation Number for Accepted Interactions**
	R1	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**	**R9**	**R10**	
1	15	16	17	17	16	17	16	16	19	20	16.9
2		22	22	23	23	23	22	24	25	27	23.4
3	23	28	29	28	31		29	31	31	34	29.3
4	28	33	36		37	33	36	40	38	39	35.5
5	33	39		41	43	40	44	48	45	48	42.3
6	38	45		45	49	47					44.8
7	46	51	58	51		59		65			55
8		57	66	58					69	70	64
9		75	72							76	74.3
10	72		81		74						75.6
	Discarded Interactions at Generation Number										**Average Generation Number for Discarded Interactions**
1											
2	19										19
3						27					0
4				34							34
5		42	42								42
6		50	50				52	56	53	56	52.8
7					54		60		62	64	60
8	54				62	69	69	72			65.2
9	64			74	68	79	82	80	75		74.5
10		88		84		84	89	89	82	84	85.7

Table C.6.Number of Interactions, Successful Interactions, and Discarded Interactions for Rosenbrock's Function n =10

Parameters	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**	**R9**	**R10**	**Average Generation Number for Interaction**
Total Interactions	10	10	10	10	10	10	10	10	10	10	10
Accepted Interactions	7	9	8	7	7	6	5	6	6	7	6.8
Discard Interactions	3	1	2	3	3	4	5	4	4	3	3.2
Population Size after Interactions	17	19	18	17	17	16	15	16	16	17	17

Table C.6: Accepted and Discarded Interactions in all Runs forRosenbrock's Function n=20

Interactions	Successful Interactions at Generation Number										Average Generation Number for Accepted Interactions
	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	
1	19	17	15	16	16	17	17	16	16	16	16.5
2	25	23	21	21	22	23	23	22	22	20	22.2
3	30	29	26	28	29	29	30	28	28	26	28.3
4	36	35	31	33	35	34	37	35	35	32	34.3
5	44	40		39	42	41	44	42		38	41.2
6			42	46			52	49	50	43	47
7	60	56	47		54		60	57	57	50	55.1
8		61	52	58			67		63		60.2
9							77		67		72
10		73	63				86	74	72		73.6
	Discarded Interactions at Generation Number										Average Generation Number for Discarded Interactions
1											0
2											0
3											0
4											0
5			37						41		39
6	53	48			49	48					49.5
7				52		54					53
8	70				60	61		63		58	62.4
9	80	67	60	64	75	66		68		65	68.1
10	87			69	82	71				79	77.6

Table C.8.Number of Interactions, Successful interactions, and Discarded Interactions for Rosenbrock's Function n =20.

Parameters	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Average Generation Number for Interactions
Total Interactions	10	10	10	10	10	10	10	10	10	10	10
Accepted Interactions	6	8	8	7	6	5	8	8	9	7	7.2
Discarded Interactions	4	2	2	3	4	5	2	2	1	3	3
Population Size after Interactions	26	28	28	27	26	25	28	28	29	27	27.2

APPENDIX D

DATASET AND USERS SPECIFICATIONS

This appendix shows more results with different users. The dataset of different L-String is also included in this appendix.

Table D.1 Dataset for L-String

1	!F[+]F[-]
2	!F[[+F[-F]]F
3	!F[-]F[+]F
4	!F[+F]F
5	!F[-F[+F]F][-F]
6	!F[+][-[+F]]
7	!F[+]FFF
8	![+]F[-]F
9	!F[^F][+F]
10	!F[+[&F]F]
11	!F[[^F]-F][+F[^F]]

Table D.2 User's specifications (First three users are undergrad students and last 2 are post graduate students)

User 1	Very new to the computer graphics and visualization, having no idea about GA. New to L-System.
User 2	Very new to the computer graphics and visualization, having no idea about GA. New to L-System.
User 3	Having idea about GA nature and optimization. Having no idea about IGA and L-System.
User 4	Well known to modelling and simulation. Knows the area of optimization.
User 5	Well known to Grammar formalism of L-system. Complete knowledge of Computer Graphics. Having no idea of IGA.

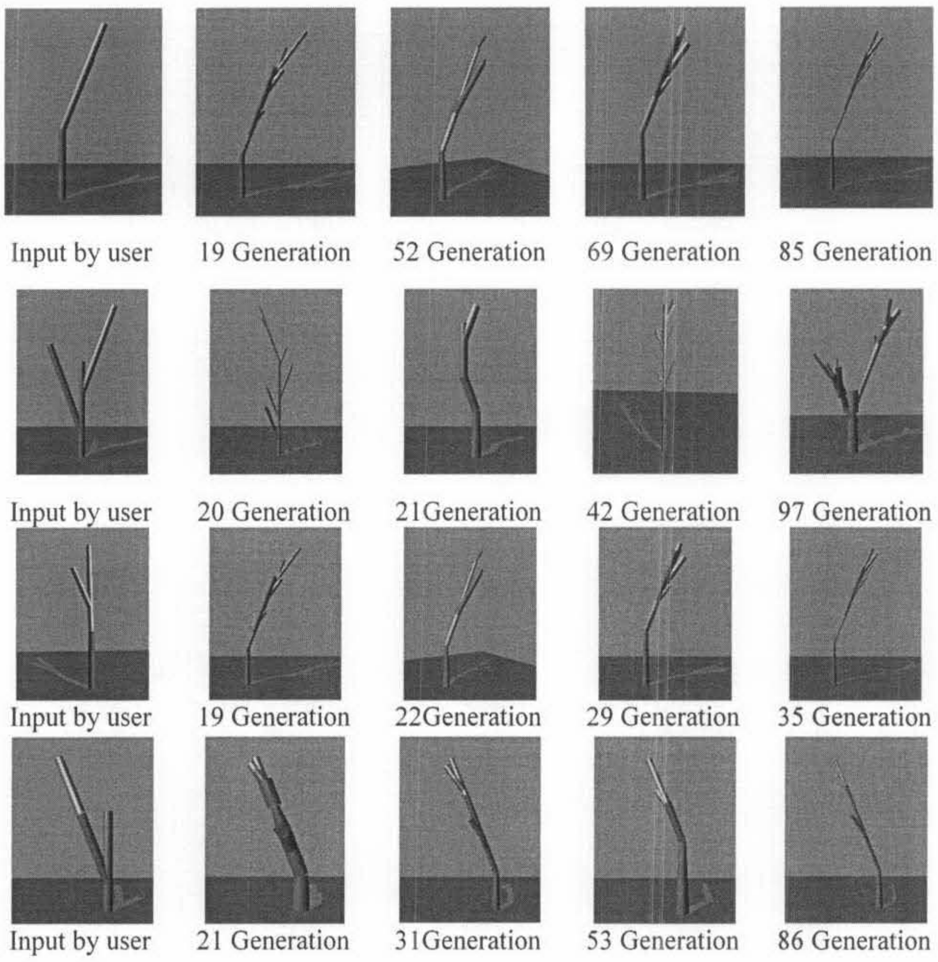


Figure D.1: Generated Structure with user perception

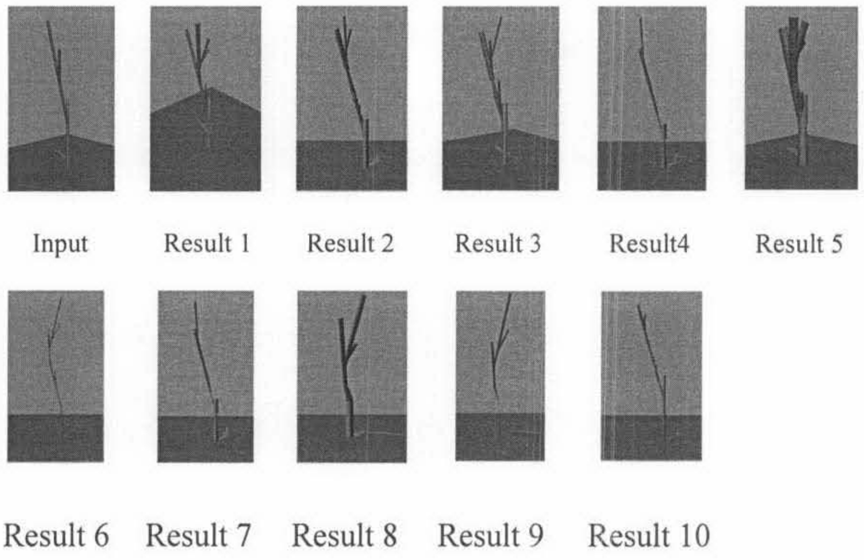


Figure D.2: Generating Structures with Constant Generation Numbers with Different Users.

Experiment 1: Result 1

Input string by user*: !F[+]F[-]*

***Input parameters** : 20, 3,26*

***L-String** : !F[+!F[+]F[-]]F[-!F[+]F[-]]*

Input to VIGA-2D

Random Generated Chromosome (n=21):

20,3,26,5,30,16,29,27,30,7,15,34,12,15,8,12,30,32,8,6,2

SGA	**VIGA-2D**
Generation 29 ***Axiom:*** *(0.24,0.6,0)A(4,6)* **Rule:** # A(x,w)->!(w)F(x)[+(20)$A (x*0.81,w*0.64)!(w*0.81)F(x*0.73)[+(20)$ A(x*0.81,w*0.61)]F(x*0.57) [-(20)$A (x*0.81,w*0.81)]]F(x*0.81)[-(20)$A(x*0.6 4,w*0.81) !(w*0.78)!(w*0.78)F(x*0.81) [+(20)$A(x*0.83,w*0.81)]F(x*0.86)[-(20)$A (x*0.55,w*0.82)]]	**Generation 29** **Axiom:** (0.24,0.6,0)A(34,5) **Rule:** # A(x,w)->!(w)F(x) [+(20)$A(x*0.81,w*0.81)!(w*0.55)F(x*0.7[) +(20)$A(x*0.81,w*0.81)] F(x*0.44)[- 20)$A (x*0.81, w*0.81]]F(x*0.81)[-(20)$A(x*0.85, w*0.51)!(w*0.81) F(x*0.81)[+(20)$A (x*0.81,w*0.66)] F(x*0.81)[- (20)$ A(x*0. 81,w*0.65)]]
Generation 59 **Axiom:** (0.24,0.6,0)A(7,3) **Rule :** # A(x,w)->!(w)F(x)[+(20)$A (x*0.42,w*0.85)!(w*0.83)F(x*0.81)[+(20)$ A(x*0.81,w*0.61)]F(x*0.76)[-(20)$A (x*0.81,w*0.81)]] F(x*0.81)[-(20) $A(x*0.64,w*0.81) !(w*0.73)F(x*0.81) [+(22)$A(x*0.81,w*0.56)]F(x*0.71)[-22) $A(x*0.68,w*0.61)]]	**Generation 59** **Axiom:** (0.24,0.6,0)A(30,4) **Rule:** # A(x,w)-!(w)F(x)[+(20) $A(x*0.69,w*0.67)!(w*0.68)F(x*0.8 5)[+(20)$A(x*0.86,w*0.36)]F(x*0.44)[-(20)$A(x*0.81,w*0.49)]]F(x*0.88) [-(20)$A(x*0.65,w*0.67)!(w*0.81) F(x*0.8[)+(20)$A(x*0.3],w*0.88)F(x *0.76)[- (20)$A(x*0.81,w*0.72)]]
Generation 75 **Axiom:** (0.24,0.6,0)A(37,7) **Rule :** # A(x,w)->!(w)F(x)[+(20)$A (x*0.42,w*0.72)!(w*0.74)F(x*0.81)1[+(20) $A(x*0.32,w*0.33)]F(x*0.53)1[-(20)$A(x*0.81,w*0.81)]]F(x*0.6 5)[-(20)$A(x*0.64,w*0.81) !(w*0.78)F(x*0.4[)+(20)$A(x*0.81,w*0.62)]F(x*0.4 7)[- 20)$A(x*0.72,w*0.42)]]	**Generation 75** **Axiom:** (0.24,0.6,0)A(37,7) **Rule:** # A(x,w)->!(w)F(x)[+ (20)$A(x*0.63,w*0.63)!(w*0.62)F(x *0.47)[+(20)$A(x*0.62,w*0.56)]F(x* 0.79)[-(20)$A(x*0.65,w*0.55)]]F(x*0 *0.55)]]F(x*0.81)[-(20)$A(x*0.48, ,w*0.51)!(w*0.77)F(x*0.67)[+(20)$A (x*0.63,w*0.61)]F(x*0.66)[-(20)$A(x*0.65,w*0.63)]]

Experiment 1: Result 2

Input string by user*:* !F[-]F[+]F

Input parameters *:* 27, 3, 21, 19

L-String *: !F[-!F[-]F[+]F]F[+!F[-]F[+]F]F*

<u>Input to VIGA-2D</u>

Randomly Generate Chromosome (n=24):

27,3,21,19,18,6,12,22,13,25,17,21,26,32,11,31,32,33,22,20,16,15,9,20

<u>SGA</u>	<u>VIGA-2D</u>
<u>Generation 30</u> ***Axiom:*** *(0.24,0.6,0)A(24,8)* **Rule:** # A(x,w)->!(w)F(x)[-(20)$A (x*0.45,w*0.45)!(w*0.81)F(x*0.52)[-(20)$A(x*0.81,w*0.86)]F(x*0.87) [+(20)$A(x*0.81,w*0.53)]F(x*0.81)]F(x*0.3[)+(20)$A(x*0.38,w*0.45)!(w*0.61)F(x*0.81)[-(20)$A(x*0.81 ,w*0.45)]F(x*0.81)[+(20)$A(x*0.45,w*0.45)]F(x*0.81)]F(x*0.81)	**<u>Generation 30</u>** ***Axiom:*** *(0.24,0.6,0)A(23,4)* **Rule:** # A(x,w)->!(w)F(x)[-(20)$A(x*0.45,w*0.45)!(w*0.35)F(x*0.73)[-(20)$A(x*0.45 ,w*0.81)]F(x*0.81)[+(20)$A(x*0.73,w*0.81)]F(x*0.37)]F(x*0.33)[+(20)$A(x*0.81,w*0.45)!(w*0.73)F(x*0.37)[-(20)$A (x*0.36,w*0.45)]F(x*0.81)[+(20)$A(x*0.45,w*0.45)]F(x*0.37)]F(x*0.81)

<u>Generation 51</u>
Axiom: (0.24,0.6,0)A(23,8)
Rule : # A(x,w)->!(w)F(x)[-(20)$A(x*0.45,w*0.45)!(w*0.67)F(x*0.8[)-(20)$A(x*0.45,w*0.73)]
F(x*0.52)[+(22)$A(x*0.46,w*0.53)]F(x*0.89)1]F(x*0.38)[+(20)$A(x*0.38,w*0.45)!(w*0.79)F(x*0.81)[-(20)$A(x*0.81,w*0.45)]F(x*0.3[)+(20)$A(x*0.45,w*0.48)]F(x*0.81)]F(x*0.81)
	<u>Generation 51</u>
	Axiom: (0.24,0.6,0)A(24,4)
	Rule: # A(x,w)->!(w)F(x)[-(20)$A(x*0.45,w*0.45)!(w*0.83)F(x*0.81)[-(20)$A(x*0.45
	,w*0.81)]F(x*0.59)[+(20)$A(x*0.42,w*0.81)]F(x*0.37)]F(x*0.33)[+(22)$A(x*0.73,w*0.73)!(w*0.71)F(x*0.8[)-(20)$A
	(x*0.45,w*0.45)]F(x*0.51)
	[+(20)$A(x*0.45,w*0.45)]F(x*0.37)1]F(x*0.81)

SGA	**VIGA-2D**
Generation 78 **Axiom:** (0.24,0.6,0)A(5,6) **Rule :** # A(x,w)->!(w)F(x)[-(20)$A (x*0.49,w*0.81)!(w*0.62)F(x*0.8[)-(20)$A(x*0.45,w*0.33)]F(x*0.52)[+(20)$A(x*0.31,w*0.51)]F(x*0.8])F(x*0.54)1[+(22)$A(x*0.54,w*0.85)!(w*0.42)F(x*0.71)[-(20)$A(x*0.45 ,w*0.45)]F(x*0.36)[+(20)$A(x*0.76,w*0.84)]F(x*0.7])F(x*0.47)	**Generation 78** **Axiom:** (0.24,0.6,0)A(18,8) **Rule:** # A(x,w)->!(w)F(x)[-(20)$A(x*0.45,w*0.57)!(w*0.47)F(x*0.68)[-(20)$A(x*0.45, w*0.45)]F(x*0.7[)+(20)$A(x*0.63,w*0.81)]F(x*0.8])F(x*0.65)[+(20)$A(x*0.54,w*0.45)!(w*0.73)F(x*0.8[)-(20)$A(x*0.45,w*0.45)]F(x*0.57)[+(20)$A(x*0.58,w*0.39)]F(x*0.44)]F(x*0.81)
Generation 100 **Axiom:** (0.24,0.6,0)A(5,6) **Rule :** # A(x,w)->!(w)F(x)[-(20)$A (x*0.85,w*0.45)!(w*0.71)F(x*0.49)[-(20)$A(x*0.45,w*0.76)]F(x*0.47) [+(20)$A(x*0.65,w*0.71)]F(x*0.35)1]F(x*0.88)[+(22)$A(x*0.54,w*0.71)!(w*0.57)F(x*0.52)[-(20)$A (x*0.45,w*0.45)]F(x*0.36)[+(20)$A(x*0.45,w*0.45)11]F(x*0.79)]F(x*0.63)	**Generation 100** **Axiom:** (0.24,0.6,0)A(26,7) **Rule:** # A(x,w)->!(w)F(x)[-(20)$A(x*0.5!,w*0.45)(w*0.82)F(x*0.42)[-(20)$A(x*0.45, w*0.51)]F(x*0.51)[+(20)$A(x*0.54,w*0.67)]F(x*0.46)]F(x*0.56)[+(20)$A(x*0.64,w*0.45)!(w*0.73)F(x*0.83)[-(20) $A(x*0.54,w*0.45)]F(x*0.55)[+(20)$A(x*0.68,w*0.46)]F(x*0.3])F(x*0.53)
Generation 110 **Axiom:** (0.24,0.6,0)A(28,6) **Rule :** # A(x,w)->!(w)F(x)[-(20)$A (x*0.85,w*0.45)!(w*0.51)F(x*0.47)[-(20)$A(x*0.45,w*0.76)]F(x*0.46) [+(20)$A(x*0.65,w*0.71)]F(x*0.67)]F(x*0.88)[+(20)$A(x*0.3!,w*0.45)(w*0.76)F(x*0.52)[-(20)$A(x*0.89 ,w*0.45)]F(x*0.51)[+(20)$A(x*0.45,w*0.45)]F(x*0.79)]F(x*0.54)	**Generation 110** **Axiom:** (0.24,0.6,0)A(31,5) **Rule** : # A(x,w)->!(w)F(x)[-(20)$A(x*0.45,w*0.45)!(w*0.68)F(x*0.58)[-(20)$A(x*0.82, w*0.51)]F(x*0.51)1[+(20)$A(x*0.54,w*0.79)]F(x*0.73)]F(x*0.76)[+(20)$A(x*0.81,w*0.45)11!(w*0.61)F(x*0.87)[-(20) $A(x*0.65,w*0.51)]F(x*0.61)[+(20)$A(x*0.45,w*0.45)]F(x*0.63)]F(x*0.81)

Experiment 1: Result 3

Input string by user*: !F[[-F]]F[+F]*

Input parameters *: 10, 7, 34, 28, 30, 31*

L-String *: !F[[-!F[[-F]]F[+F]F]]F[+!F[[-F]]F[+F]F]*

<u>**VIGA-2D**</u>

Random Generated Chromosome (n= 27):

10,7,34,28,30,15,13,13,29,21,23,30,14,34,20,19,17,22,25,31,13,9,11,19,15,5,11

<u>**SGA**</u>	<u>**VIGA-2D**</u>
<u>**Generation 31**</u> ***Axiom:*** *(0.24,0.6,0)A(12,8)* **Rule:** # A(x,w)->!(w)F(x)[[-(20)$A(x*0.45,w*0.45)!(w*0.43)F(x*0.57)[[-(20)$A(x*0.81,w*0.45) F(x*0.81)1]]F(x*0.42)[+(20)$A(x*0.54,w*0.81)F(x*0.63)]F(x*0.78)1]]F(x*0.81)[+(20)$A(x*0.45,w*0.45)!(w*0.71)F(x*0.65)[[-(20)$A(x*0.45, w*0.81)F(x*0.65)]]F(x*0.64)[+(20)$A(x*0.73,w*0.81)F(x*0.81)1]F(x*0.82)]	<u>**Generation 31**</u> ***Axiom:*** *(0.24,0.6,0)A(33,4)* **Rule:** # A(x,w)->!(w)F(x)[[-(20)$A(x*0.45,w*0.45)!(w*0.83)F(x*0.59)[[-(20)$A(x*0.73,w w*0.51)F(x*0.81)]]F(x*0.81)[+(20)$A(x*0.45,w*0.49)F(x*0.61)]F(x*0.44)]]F(x*0.39)[+(20)$A(x*0.45,w*0.81)!(w*0.59)F(x*0.81)[[-(20)$A(x*0.7F,w*0. .57)(x*0.66)1]]F(x*0.32)[+(20)$A(x*0.65,w*0.81)F(x*0.81)]F(x*0.6])
<u>**42 Generation**</u> **Axiom:** (0.24,0.6,0)A(38,4) **Rule :** # A(x,w)->!(w)F(x)[[-(20)$A (x*0.51,w*0.45)!(w*0.51)F(x*0.56)[[-(20)$A(x*0.81,w*0.45)F(x*0.57)]]F(x*0.82)[+(20)$A(x*0.35,w*0.45)F(x*0.82)]F(x*0.78)1]]F(x*0.81)[+(20)$A(x*0.45,w*0.85)!(w*0.38)F(x*0.5[)[-(20)$A(x*0.45,w*0.81) F(x*0.79)]]F(x*0.64)[+(20)$A(x*0.88,w*0.81)F(x*0.81)]F(x*0.82)]	<u>**42 Generation**</u> **Axiom:** (0.24,0.6,0)A(35,3) **Rule:** # A(x,w)->!(w)F(x)[[-(20)$A(x*0.45,w*0.45)!(w*0.49)F(x*0.51)[[-(20)$A(x*0.81,w *0.75)F(x*0.64)]]F(x*0.59)[+(20)$A(x*0.45,w*0.45)F(x*0.55)]F(x*0.54)]]F(x*0.52)[+(20)$A(x*0.45,w*0.81)!(w*0.61)F(x*0.56)[[-(20)$A(x*0.31,w*0.4 5)F(x*0.61)]]F(x*0.75)[+(20)$A(x*0.81,w*0.45)F(x*0.61)]F(x*0.86)]

SGA	VIGA-2D
Generation 62 **Axiom:** (0.24,0.6,0)A(30,4) **Rule** : # A(x,w)->!(w)F(x)[[-(20)$A(x*0.85,w*0.45)!(w*0.51)F(x*0.55)[[-(20)$A(x*0.45,w*0.45) F(x*0.8])]F(x*0.45)[+(20)$A(x*0.35,w*0.45)F(x*0.31)]F(x*0.41)]]F(x*0.81)[+(20)$A(x*0.45,w*0.45)!(w*0.36)F(x*0.65)[[-(20)$A(x*0.45, w*0.45)F(x*0.73)]]F(x*0.64)[+(20)$A(x*0.45,w*0.69)F(x*0.81)]F(x*0.82)]	**Generation 62** **Axiom:** (0.24,0.6,0)A(34,2) **Rule:** # A(x,w)->!(w)F(x)[[-(20)$A(x*0.45,w*0.45)!(w*0.63)F(x*0.64)[[-(20)$A(x*0.81, w*0.45)F(x*0.62)]]F(x*0.46)[+(20)$A(x*0.69,w*0.82)F(x*0.71)]F(x*0.71)]]F(x*0.54)[+(20)$A(x*0.45,w*0.45)!(w*0.64)F(x*0.81)[[-(20)$A(x*0.31,w*0. 57)F(x*0.73)]]F(x*0.38)[+(20)$A(x*0.45,w*0.45)F(x*0.81)]F(x*0.77)]
Generation 77 **Axiom:** (0.24,0.6,0)A(37,3) **Rule** : # A(x,w)->!(w)F(x)[[-(20) $A(x*0.31,w*0.45)!(w*0.54)F(x*0.32)[[-(20)$A(x*0.45,w*0.45)F(x*0. 58)]]F(x*0.51)[+(20)$A(x*0.45,w*0.72)F(x*0.65)]F(x*0.78)]]F(x*0.74)[+(20)$A(x*0.45,w*0.45)!(w*0.59)F(x*0.65)[[-(20)$A(x*0.45,w*0.45) F(x*0.73)]]F(x*0.64)[+(20)$A(x*0.45,w*0.45)F(x*0.78)]F(x*0.59)]	**Generation 77** **Axiom:** (0.24,0.6,0)A(38,3) **Rule:** # A(x,w)->!(w)F(x)[[-(20)$A(x*0.55,w*0.62)!(w*0.72)F(x*0.58)[[-(20)$A(x*0.45 ,w*0.45)F(x*0.35)1]]F(x*0.37)[+(20)$A(x*0.45,w*0.45)F(x*0.61)]F(x*0.76)]]F(x*0.32)[+(20)$A(x*0.79,w*0.45)!(w*0.64)F(x*0.58)[[-(20)$A(x*0.45, w*0.45)F(x*0.68)]]F(x*0.3[)+(20)$A(x*0.72,w*0.45)F(x*0.81)]F(x*0.75)]

Experiment 1: Result 4

Input string by user*:* !F[-F[+F]F][-F]

Input parameters *: 27, 3, 34, 11, 22, 33*

L-String *: !F[-F[+F]F][-F]*

VIGA-2D

Random Generated Chromosome (n= 12):

27,3,34,11,22,33,17,28,5,32,6,14

<u>**SGA**</u>	<u>**VIGA-2D**</u>
<u>**Generation 19**</u> ***Axiom:*** *(0.24,0.6,0)A(15,7)* **Rule:** # A(x,w)->!(w)F(x)[-(20)$A(x*0.81,w*0.81)F(x*0.81)[+(20)$A(x*0.5F,w*0.81)1(x*0.81)]F(x*0.81)][-(20)$A(x*0.81,w*0.45) F(x*0.89)]	<u>**Generation 19**</u> ***Axiom:*** *(0.24,0.6,0)A(29,4)* **Rule:** # A(x,w)->!(w)F(x)[-(20)$A(x*0.44,w*0.73)F(x*0.81) [+(20)$A (x*0.81,w*0.76) F(x*0.81)]F(x*0.81)][-(20)$A(x*0.82,w*0.81)F(x*0.81)]
<u>**Generations 52**</u> **Axiom:** (0.24,0.6,0)A(8,4) **Rule :** # A(x,w)->!(w)F(x)[-(20)$A (x*0.35,w*0.79)F(x*0.31)[+(20)$A((x*0.37,w*0.81)11F(x*0.62)]F(x*0.81)][-(20)$A(x*0.83,w*0.88) F(x*0.89)]	<u>**Generation 52**</u> **Axiom:** (0.24,0.6,0)A(17,3) **Rule:** # A(x,w)->!(w)F(x)[-(20)$A(x*0.84,w*0.56)F(x*0.81) [+(20)$A(x*0.5F ,w*0.56) (x*0.43)]F(x*0.65)][-(20)$A(x*0.81,w*0.79)F(x*0.31)]
<u>**Generation 69**</u> **Axiom:** (0.24,0.6,0)A(33,6) **Rule** : # A(x,w)->!(w)F(x)[-(20)$A(x*0.35,w*0.51)F(x*0.3[)+(20)$A(x*0.37,w*0.81)F(x*0.76)]F(x*0.4])[-(20)$A(x*0.45,w*0.47) F(x*0.48)]	<u>**Generation 69**</u> **Axiom:** (0.24,0.6,0)A(23,3) **Rule:** # A(x,w)->!(w)F(x)[-(20)$A(x*0.5F,w*0. 5)(x*0.61) [+(20)$A (x*0.49 ,w*0.76) F(x*0.81)]F(x*0.75)][- (20)$A(x*0.81,w*0.79) F(x*0.84)]
<u>**Generation 85**</u> **Axiom:** (0.24,0.6,0)A(29,3) **Rule:**# A(x,w)->!(w)F(x)[-(20)$A(x*0.35,w*0.51)F(x*0.3[)+(20)$A(x*0.37,w*0.81)F(x*0.76)]F(x*0.4])[-(20)$A(x*0.45,w*0.47) F(x*0.48)]	<u>**Generation 85**</u> **Axiom:** (0.24,0.6,0)A(29,3) **Rule:** # A(x,w)->!(w)F(x)[-(20)$A(x*0.71,w*0.44)F(x*0.73)[+(20)$A(x*0.44, w*0.79) F(x*0.81)]F(x*0.71)][- (22)$A (x*0.81,w*0. 31)F(x*0.67)]

Experiment 1: Result 5

Input string by user*: !F[F][&]F*

Input parameters *: 12, 5, 10, 15*

L-String *: !F[F][&!F[F][&!F[F][&]F]F]F*

VIGA-2D

Random Generated Chromosome (n= 18):

12,5,10,15,19,29,12,25,19,22,10,19,24,6,19,8,28,33

SGA	**VIGA-2D**
Generation 23 ***Axiom:*** *(0.24,0.6,0)A(4,8)* **Rule:** # A(x,w)->!(w)F(x)[F(x* 0.37)][&(20)$A(x*0.81,w*0.81)!(w*0.81)F(x*0.81)[F(x*0.3])[&(20)$A(x*0.81,w*0.68)!(w*0.81)F(x*0.68)[F(x*0.79)][&(20)$A(x*0.45,w*0.45)]F(x*0.81)]F(x*0.81)1]F(x*0.84)	**Generation 23** ***Axiom:*** *(0.24,0.6,0)A(35,4)* **Rule:** # A(x,w)->!(w)F(x)[F(x* 0.58)][&(20)$A(x*0.59,w*0.58)!(w*0.55)F(x*0.56)1[F(x*0.52)][&(20)$A(x*0.55,w*0.49)!(w*0.61)F(x*0.55)1[F(x*0.47)][&(20)$A(x*0.55,w*0.42)]F(x*0.68)1]F(x*0.61)]F(x*0.51)
Generation 37 **Axiom:** (0.24,0.6,0)A(21,7) **Rule :** # A(x,w)->!(w)F(x)[F(x* 0.57)][&(20)$A(x*0.81,w*0.81)!(w*0.85)F(x*0.81)[F(x*0.3])[&(20)$A(x*0.81,w*0.45)!(w*0.57)F(x*0.68)[F(x*0.55)][&(20)$A(x*0.45,w*0.45)]F(x*0.81)]F(x*0.81)1]F(x*0.47)	**Generation 37** **Axiom:** (0.24,0.6,0)A(32,6) **Rule:** # A(x,w)->!(w)F(x)[F (x*0.8])[&(20)$A(x*0.59,w*0.57)!(w*0.61)F(x*0.55)[F(x*0.53)][&(20)$A(x*0.55,w*0.65)!(w*0.68)F(x*0.81)[F(x*0.79)][&(20)$A(x*0.45,w*0.54)]F(x*0.39)]F(x*0.81)]F(x*0.42)
Generation 75 **Axiom:** (0.24,0.6,0)A(12,4) **Rule :** # A(x,w)->!(w)F(x)[F(x* 0.87)][&(20)$A(x*0.41,w*0.81)!(w*0.72)F(x*0.3[)F(x*0.57)][&(20)$A(x*0.33,w*0.45)!(w*0.41)F(x*0.72)1[F(x*0.88)][&(20)$A(x*0.45,w*0.45)]F(x*0.84)]F(x*0.74)]F(x*0.48)	**Generation 75** **Axiom:** (0.24,0.6,0)A(30,5) **Rule:** # A(x,w)->!(w)F(x)[F(x *0.57)][&(20)$A(x*0.51,w*0.53)!(w*0.83)F(x*0.58)[F(x*0.64)][&(20)$A(x*0.48,w*0.45)!(w*0.81)F(x*0.65)[F(x*0.31)][&(20)$A(x*0.45,w*0.84)]F(x*0.76)]F(x*0.88)]F(x*0.76)

Experiment 2

Input by User : !F[[-F]]F[+F]

Input Parameters: *11, 7, 29, 28, 26, 7*

L-String : !F[[-!F[[-F]]F[+F]F]]F[+!F[[-F]]F[+F]F]

VIGA-2D for all results.

Random Generated Chromosome (n=27):

,11,7,29,28,26,23,22,23,32,12,5,32,30,31,33,23,28,10,29,23,29,12,22,33,25,34,29

Result 1:

Axiom: *(0.24,0.6,0)A(31,4)*

Rule: *# A(x,w)->!(w)F(x)[[-(20)$A(x*0.45,w*0.45)!(w*0.49)F(x*0.8*

*1)[[-(20)$A(x*0.45,w*0.48)F(x*0.38)]]F(x*0.47)[+(20)$A(x*0.63,w**

*0.35)F(x*0.81)]F(x*0.44)]]F(x*0.5[)+(20)$A(x*0.45,w*0.45)!(w*0.41)F(x*0.46)[[-(20)$A(x*0.72,w*0.45)F(x*0.82)1]]F(x*0.47)[+(20)$A*

*(x*0.47,w*0.45)F(x*0.67)1]F(x*0.62)]*

Result 2:

Axiom: *(0.24,0.6,0)A(31,7)*

Rule: # A(x,w)->!(w)F(x)[[-(20)$A(x*0.45,w*0.45)!(w*0.83)F(x*0.8

7)[[-(20)$A(x*0.61,w*0.64)F(x*0.52)1]]F(x*0.63)[+(20)$A(x*0.81,

w*0.57)11F(x*0.83)]F(x*0.5])]F(x*0.67)[+(20)$A(x*0.45,w*0.45)!(w*0.66)F(x*0.6[)[-(20)$A(x*0.45,w*0.45)F(x*0.58)1]]F(x*0.65)[+(2

0)$A(x*0.65,w*0.45)F(x*0.46)1]F(x*0.87)]

Result 3:

Axiom: *(0.24,0.6,0)A(37,8)*

Rule: *# A(x,w)->!(w)F(x)[[-(20)$A(x*0.77,w*0.45)!(w*0.56)F(x*0.7*

*8)[[-(20)$A(x*0.45,w*0.45)F(x*0.81)]]F(x*0.49)[+(20)$A(x*0.89,w**

*0.45)F(x*0.36)1]F(x*0.51)]]F(x*0.81)[+(20)$A(x*0.45,w*0.45)!(w*0.32)F(x*0.81) [[-(20)$A(x*0.4F,w*0.81)(x*0.53)]] F(x*0.81)[+(20)$A*

*(x*0.45,w*0.45)F(x*0.69)1]F(x*0.61)]*

Result 4:

Axiom: *(0.24,0.6,0)A(19,3)*

Rule: *# A(x,w)->!(w)F(x)[[-(20)$A(x*0.74,w*0.51) !(w*0.32)F(x*0. 79)[[-(20)$A(x*0.45,w*0.67)F(x*0.41)]]F(x*0.89)[+(20)$A(x*0.45,*

*w*0.45)F(x*0.84)]F(x*0.84)1]]F(x*0.31)[+(20)$A(x*0.45,w*0.39)!(w*0.81)F(x*0.81)[[-(20)$A(x*0.55,w*0.61)F(x*0.86)]]F(x*0.81)[+(2*

*0)$A(x*0.86,w*0.45)F(x*0.51)1]F(x*0.55)]*

<u>Result 5:</u>

Axiom: *(0.24,0.6,0)A(28,8)*

Rule:
```
# A(x,w)->!(w)F(x)[[-(20)$A(x*0.81,w*0.45)!(w*0.77)F(x*0.72)[[-
(20)$A(x*0.66,w*0.45)F(x*0.58)]]F(x*0.71)[+(20)$A(x*0.45,w*0.45)F(x*0.68)]F(
x*0.81)]]F(x*0.81)[+(20)$A(x*0.5!,w*0.45)(w*0.77)F(x*0.82)[[-(20)$A(x*0.45
,w*0.45)F(x*0.81)1]]F(x*0.76)[+(20)$A(x*0.85,w*0.45)F(x*0.36)]F(x*0.62)]
```