Interactive Evolutionary 3D Fractal Modeling

PANG, Wenjun

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Philosophy

in

Mechanical and Automation Engineering

The Chinese University of Hong Kong
September 2009

Thesis / Assessment Committee

Professor Ronald Chung (Chair)

Professor K.C. Hui (Thesis Supervisor)

Professor Charlie C.L. Wang (Committee Member)

ACKNOWLEDGEMENTS

My sincere appreciation goes to my supervisor, Professor Kinchuen Hui, who continuously orients my research towards the right direction, provides full academic and spiritual support and corrects my English in thesis writing. Without his constant guidance and encouragement, this work would not have been finished successfully.

I would also express my deep gratitude to Professor Charlie C. L. Wang for his valuable advices and support to my research.

All the fellow members in Computer-Aided Design Lab and staffs in the Department General Office deserve my appreciation for making my life easier and happier during my study. I would like to express special thanks to Wang Chengdong, whose selfless help and encouragement give me so much confidence in my life and research. I am thankful to Yuki for helping me with her ideas that improved my understanding of the scope of my work.

ABSTRACT

Research on art fractal has captured wide attention and gained considerable achievement in the past two decades. Most related works focus on developing two dimensional fractal art, and the fractal art tools usually just assist in the creation process, but cannot perform an automatic generation associated with aesthetic evaluation. The percentage of visually attractive fractals generated by fractal art construction formula, such as Iterated Function Systems, is not high. It is thus essential to develop efficient techniques for automatic 3D art fractals generation with user interaction.

This paper presents a technique to create 3D fractal art forms automatically, by which designers can get access to a large number of 3D art shapes that can be modified interactively. This is based on a modified evolutionary algorithm using a Fractal Transformation (FT) Iterated Function System, which provides tunable geometric parameters. Fitness function sorting the fractal aesthetic value applied in evolutionary system is formulated based on characteristic parameters in fractal theory, including capacity dimension, correlation dimension and largest Lyapunov exponent. The productivity of visually appealing fractal can be enhanced greatly by using proposed technique. Experiments demonstrated the effectiveness of the proposed FT IFS formula and evolutionary system. The proposed technique can be applied to design jewelry, light fixture and decoration, and corresponding examples are included.

摘 要

在過去二十年，對分形藝術的研究已經取得了客觀的進展并獲得廣泛的關注。大部分的相關工作主要致力于開發二維分形藝術，分形藝術工具通常只能輔助創造過程，但不能自動生成分形體并評價其美學價值。現在的分形藝術構造公式，例如迭代函数系统，創建的分形體中美學價值高的分形體所占比例并不高。因此研究自動生成三維分形體并允許用戶交互修改的技術是十分有必要的。

這篇論文展示了一個自動創造三維藝術分形體的方法，藉由這個方法設計師可以獲得大量的三維藝術體并交互地修改它們。這個方法包括一個能夠提供可調幾何參數的分形變換迭代函数系统 (FT IFS)，同時采用了改進的進化算法。進化系統中應用的衡量分形體美學價值的適函數是由分形特徵參數制定的，這些參數包括容量維數，關聯維數以及最大李亞普諾夫指數。這種方法大大提升了所創建的分形體中美學價值高的分形體所占比例。實驗證明了本論文提出的 FT IFS 方法的有效性。本論文提出的方法可以應用在珠寶，燈飾及裝飾品設計上，文章最後包含了相應的實驗案例。

CONTENTS

List of Tables

List of Figures

1. INTRODUCTION

Fractals are applied to art and design for over 20 years. One pioneer work is published in an article about the Mandelbrot Set published in "Scientific American" in 1985 [53]. Since then, many advances have been made, both in fractal rendering capabilities and in the understanding of fractal geometry. The past decade has already seen incredible evolution, especially in the development of two dimensional fractal images, and the corresponding computer aided design tools have become much more sophisticated. Fractal images typically are manifested as prints, bringing Fractal Artists into the company of painters and photographers. Fractals exist natively as electronic images. This is a format that traditional visual artists are quickly embracing, bringing them into fractal art's digital realm.

Fractal Art (FA) is a genre concerned with fractals—shapes or sets characterized by self affinity and an infinite amount of detail, at all scales. It is not only a tool for the expression of visual ideas, but also a visual depiction of complex mathematical equations. Fractals art works are typically created by calculating fractal objects using an iterative numerical process on a digital computer, and the computer-aided random selection of parameters in the fractal generation rule brings infinite artistic creations and unconventional patterns to artists. The calculation results are represented as still images, animations, music, textures etc.. For the most cases, it is created indirectly with the assistance of fractal generating software, iterating through three phases: setting parameters of appropriate fractal, executing the possibly lengthy calculation and evaluating the result. Examples of fractal art works are shown in Figure 1.1.

Fractals Arts are sometimes combined with human-assisted evolutionary algorithms, either by iteratively choosing good-looking specimens in a set of random variations of a fractal artwork to produce new variations or collectively like Electric Sheep project. In Electric Sheep project, people use fractal flames rendered with distributed computing as their screensaver and rate the flame they are viewing, and the rating can influence the server to reduce the traits of the undesirables and increase those of the desirables to produce a computer-generated, community-created piece of art. This method can avoid dealing with cumbersome or unpredictable parameters.

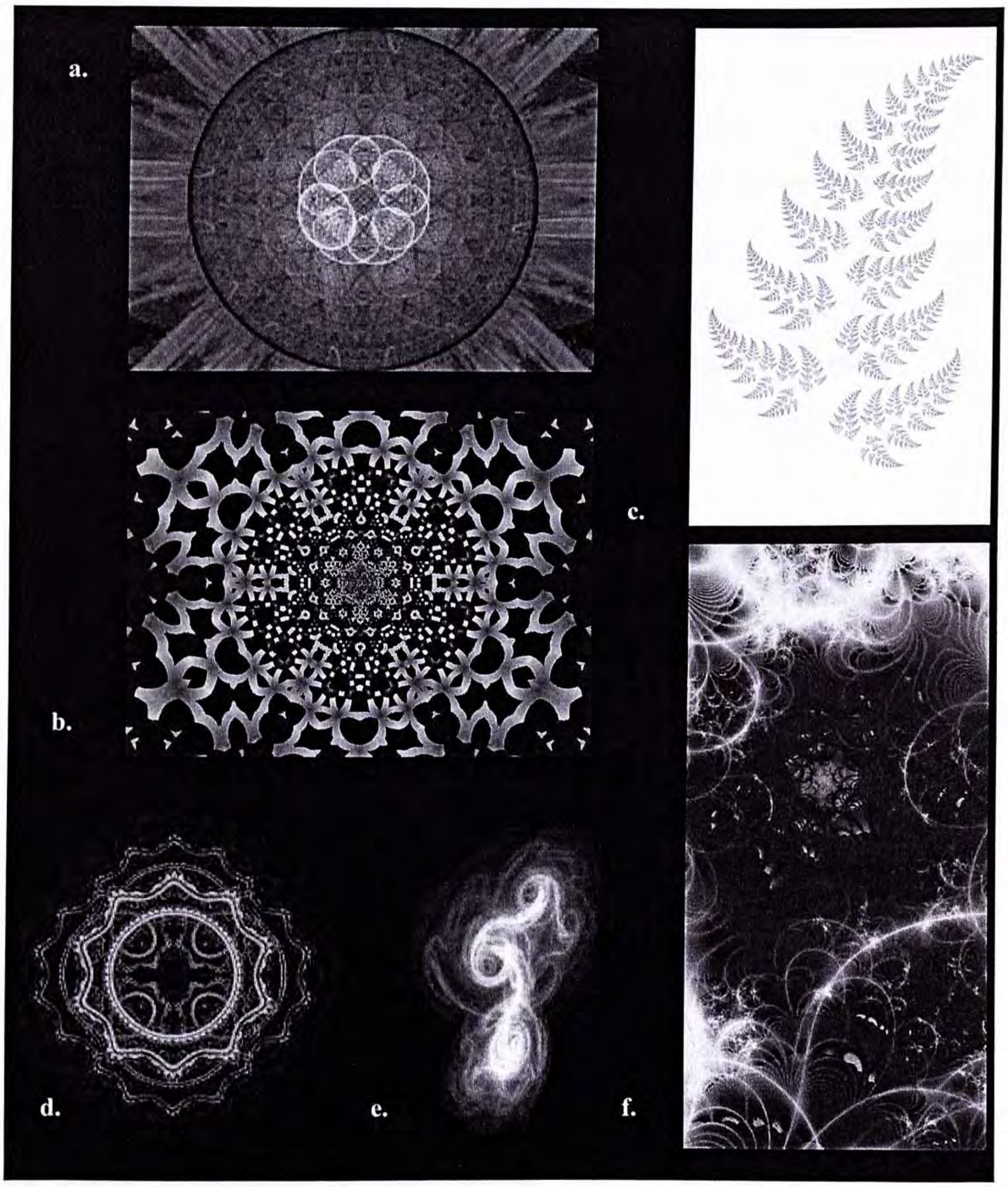


Figure 1.1: Collection of fractal art works

a) A fractal flame created by the Electric Sheep [19]; b) Example of Fractal Art by Karl Scherer, calculated with Fractint [35] c) Bransleys Fractal Fern[41]; d) A fractal created using the program Apophysis and a julian transform [41]; e) Fractal flame named 191 [37]; f) A non-Mandelbrot sterling fractal. [30].

As we can see from the examples in figure 1.1, FA is commonly for two dimensional visual art at present, and is in many respects similar to photography. Most research and applications are confined to 2D image with limited applications in image compression, computer graphics, and education. It would be beneficial to extend fractals art to 3d and apply it to design and manufacturing industry.

Next generation of CAD will be Computer-Automation Design rather than Computer-Aided Design. The current creation process of fractal art requires knowledge of fractal theory, and most fractal art software just assists in the creation process, but can not perform automatic generation with aesthetic evaluation.

Iterated Function System, introduced by M. Barnsley and S. Demko [2], is a unified way of generating a broad class of fractals and has been widely used. IFS is further employed to build artistic fractals and evaluated the aesthetic appeal with fractal characteristics [43], and a series of related research and art works emerged subsequently. However, the percentage of visually attractive fractals generated by IFS associated with broadly used Random Iteration Algorithm is very low, and the parameters controlling the process cannot be used for interactively modifying the generated fractal pattern. Evolution system with certain selection criteria and user interaction might be a possible way to solve this problem.

1.1 Recent research work

Our interactive evolutionary 3D fractal modeling work is based on the fractal construction and IFS formula, experimental aesthetics evaluation, and the evolutionary design theory, the related research works are examined in this section.

1.1.1 Fractal Art and the IFS

Fractals are applied to art and design for long time, and the usual form of expression are still images, animations, music, textures etc.. They are generally divided into four main categories: escape time fractals, Lindenmayer systems, stochastic synthesis and Iterated function systems. IFS is most widely used by artist to create 2D still fractal images. The generated fractal images can be viewed as art collection, texture and screen saver, and its application are investigated by many researchers. Scott Draves' fractal flame uses a two-dimensional IFS to create images by plotting the output of a chaotic attractor on the image plane with non-linear functions, log-density display, and structural coloring [37]. Based on the fractal flame, Electric Sheep are put forward to automatically generate distributed screen-saver using genetic algorithm [36]. More recently, fractal is also applied to design jewelry. Wannarumon Somlak etc. put forward an aesthetic-driven evolutionary approach to create two dimensional art forms for user-centered jewelry design [50][51]. K.M. Yu et al. proposed a RAT data structure to represent 2d IFS fractal curves and rapid prototyping them [56]. Later K.M. Yu et al. put forward a RBT data structure to represent 3D IFS fractal [57]. W.J. Pang and K.C. Hui presented a technique for the automatic generation of 3D art form, which allows designers to get access to large number of 3D shapes that can be altered interactively [58]. There are various applications of fractals especially in art and aesthetic field; however, research on fractals applied to three dimension is still rare. Berkowitz J. has tried some 3D fractal generation in Fractal Cosmos and uses them as virtual scene [7].

Iterated Function System, which was introduced in [2], allows effective modeling of a large class of complex objects with unlimited detail. It also provides a very compact representation, efficient computation, and a very small amount of user specification. In addition, IFS has the advantage of a single specification method to obtain a very large class of fractals, where "class" refers to a significant difference in subjective visual properties. Though IFS fractals can be of any number of dimensions theoretically, they are commonly computed and drawn in 2D.

1.1.2 Aesthetic Evaluation

Aesthetics is the science of "beautiful." Literally, it means the "science of the senses" and is concerned with sensuous perception and its realizations. Aesthetics and aesthetic measure have been researched in various ways long time ago. Galileo's father performed experiments on the aesthetics of musical intervals according to different musical scales, or tunings, published in 1588. Fechner's investigations prove that the measurements that reflect golden ratios are the most satisfying to men's eyes. From his surveys, the golden rectangles were chosen preferentially by over 75% of participants [13]. Moles presents a remarkable possibility to apply information theory to the study of aesthetic perception. He examines and analyzes the formal distinction between semantic and aesthetic information. Semantic information is the message contained in sequence. Aesthetic information is sensory, and restricted to the preferential choices of individuals [27].

Around the 1970s, Daniel Berlyne created the field of experimental aesthetics and examined the relations between verbal evaluations and exploratory behavior, and further applied the method to various fields, such as explaining music listening behavior and musical preference [29][8]. Remko and Rens review several aesthetic measure theories. They suggest that building formal models of human perceptual processes are the basis of any empirical aesthetic measure [32]. Staudek studies visual patterns and perceptions. He presents a system with algorithmic aesthetics, which integrates a computer into artistic creation and aesthetic evaluation. His work involves mathematics, geometry, perceptual psychology, theory of communication, and computer graphics to classify and assess aesthetics. His system can generate algorithmic arts from a set of abstract images, textures, and patterns. Aesthetic functions evaluate aesthetics in terms of order, complexity, harmony, variety, entropy, and redundancy [45].

Since fractals are applied to art and design field, investigations on the visual aesthetics of fractals gradually come out, and the compactness, connectivity, regularity, and symmetry of fractals were taken into considerations. Mandelbrot's work brought attention to the relationship of fractal mathematics and dynamic systems to the field of aesthetics [24][54]. J.C. Sprott proved that there is a relationship between aesthetic judgments of fractal and their characteristic parameters, and a Gaussian-format formula is used to evaluate the fitness function with two parameters [43]. Sprott and Aks researched on the effect of Lyaponov exponent (quantifying the dynamics that produce fractal patterns) on visual appeal [1]. Spehar and Minita etc. demonstrated that the value of aesthetics, the compactness and connectivity of fractals mostly depends on the fractal dimensions and the Lyapunov exponent [26][40]. Galanter & Levy suggested that complex forms and structures reflect aesthetic expression via emergent organizing properties, self-organizing behavior, and chaotic dynamic [14].

1.1.3 Evolutionary Design

Since its introduction in the mid 1960's, evolutionary algorithms have provided many novel and interesting solutions to problems in a wide variety of domains. The use of evolutionary algorithms to generate designs has attached much attention over the last twenty years. Evolutionary designs and art is rooted in mimicking natural evolution which is the survival of the fittest. In evolutionary art system, evolution acts as a form generator and can provide designer with much more design alternatives.

There has been some research in the applications of evolutionary design for the creations of artistic images and forms (e.g., Bentley & Wakefield [4]; Gero [15];

Rosenman [33]; Koile [21]; Poirson et al. [31]). Todd & Latham use evolutionary design to build sphere and ellipsoid to form ribs, horns and mathematical shapes with application in images, textures, sculptures, and animations [47]. Bentley P, Todd S. and Latham W. [6][46] applied evolution method to the design process, such as art and aesthetic forms generation. Eckert et al. [10] put forward Eckert's garment shape design system for modeling 2D garment parts based on the evolutionary design system in garment and knitwear design industry. Rowland & Biocca [34] used recursive tree to sculpture graph evolutionally for modeling 3D human heads and abstract forms. Cho [9] applied evolutionary algorithm to create 3D arm and sleeve part, neck and body part, and skirt and waistline part forms in fashion apparel design. Unemi [48] produces expression-based images and short musical pieces by evolution method.

In general, evolutionary art is created with the algorithms that produce infinite alternatives, and which inherit high-quality characteristics from the existing ones and preserve variability of alternatives at the same time. Therefore, designers are able to explore more alternatives according to their own preferences.

1.2 Objectives

The objective of this thesis is to develop a technique that builds three dimensional fractals automatically and effectively by evolutionary system, while allowing for intuitive control on amending the pattern of the fractals as well as an effective means for evaluating the aesthetics of the generated three dimensional fractals.

While many researchers and artists studied the generation of fractals, they largely focus on two dimensional images and the efficiency of fractal generation is always low. Although aesthetics evaluation is usually subjective, algorithmic aesthetics evaluation is essential for the evolutionary generation process.

In addition, the representation and visualization of 3D fractals, and its applications in design are also a major concern in this thesis.

1.3 Thesis Organization

Chapter 1 is the introduction to this thesis, and provides a summary of the background of this thesis and recent related research work. The task description and the thesis organization are also provided.

Chapter 2 provides a detailed study of fractal modeling. The most popular and sound fractal constructing method, Iterated Function System, and its mathematical foundation are presented as central theme. The fractal measurement and its aesthetic evaluation are also introduced in this chapter.

Chapter 3 focuses on the theory and algorithm of evolutionary design, including how to initialize evolution, select parents and reproduce offspring and the termination rule. Two commonly used reproduction operations, crossover and mutation are discussed. The aim of this chapter is to give an introduction to evolutionary design, based on which we employ a revised evolutionary algorithm is proposed, and which is discussed in the next chapter.

Chapter 4 studies how to build fractals with good visual aesthetics while allowing for intuitive control. A revised IFS formula, called Fractal Transformation IFS (FT IFS), together with the modified evolutionary system is suggested for building fractals based on the mathematics introduced in Chapter 2 and 3. This chapter mainly focuses on formulating the FT IFS under self-similar condition of fractals, the way to map the genotype to phenotype by a revised random iteration algorithm. This chapter also discusses the method to decode the parameters by FT IFS formula to fractal genotype, and the way these genotypes evolve and generate visually pleasing fractals by the proposed evolution system. The transforming property of the FT IFS and its effect on user interaction is explored. The fitness function of evolutionary algorithm is discussed and the details are further explained in the following chapter.

Chapter 5 is devoted to the fractal fitness function formulation. Gaussian function and the characteristic parameters of fractal, such as the capacity dimension, correlation dimension and largest Lyapunov exponent, are employed to build the fitness function. The meaning and calculation method of these fractal characteristic parameters are revealed. Correlation analysis and linear regression are used to analyze thousands of cases for studying the relationships between fractal parameters and the fitness. The normalized fitness function is then brought forward.

Chapter 6 reveals the values of the parameters employed in the experiment and their results. As shown by the results of experiments, the average fitness value and the excellent rate of fractals (Excellent fractals refers to those fitness value above 0.85) both escalate from original IFS to FT IFS with evolutionary system, which demonstrates the efficiency and effectiveness of proposed technique.

Chapter 7 is concerned with the visualization of fractal phenotype and its applications. Voxels are used as the rendering primitives for coarse rendering effect. Mesh surfaces of the fractals are reconstructed by Marching Cubes to obtain higher quality rendering. Coloring schemes, including linear gradient coloring and dual-tone coloring, are put forward. A collection of visually appealing fractals is exhibited.

Chapter 8 states the conclusions drawn from this research and suggests possible directions for future research.

2. FRACTAL MODELING

Fractal is a self-similar structure whose geometrical and topographical features are recapitulated in miniature on finer and finer scales. The major theories and concepts of fractals used in the research are briefly introduced in this chapter, including the definition of fractals, the applications of the fractals, the mathematic and geometric foundation of fractals, their construction, measurement and aesthetics.

2.1 Fractal and Fractal Art

2.1.1 Fractal

Since the 1960s, Benoit Mandelbrot started investigating self-similarity which were built on the earlier work by Lewis Fry Richardson. In 1975 Mandelbrot coined the word "fractal" to denote an object whose Hausdorff-Besicovitch dimension is greater than its topological dimension. He illustrated this mathematical definition with striking computer-constructed visualizations. These images captured the popular imagination; many of them were based on recursion, leading to the popular meaning of the term "fractal". By the point of fractal view, people may see things differently. "You risk the loss of your childhood vision of clouds, forests, galaxies, leaves, feathers, flowers, rocks, mountains, torrents of water, carpets, bricks, and much else besides. Never again will your interpretation of these things be quite the same."[3]

Fractal is a language for describing nature. Nature displays self-similar structures over an extended, but finite, scale range. Examples include clouds, lightning bolts, snow flakes, crystals, mountain ranges, river networks, cauliflower or broccoli, and systems of blood vessels and pulmonary vessels. Coastlines may be loosely considered as fractal in nature. And not all self-similar objects can be described with fractals—for example, the real line (a straight Euclidean line) is formally self-similar but fails to have other fractal characteristics; for instance, it is regular enough to be described in Euclidean terms.

A fractal is a complex shape which, when viewed in finer and finer detail, shows itself to be constructed of ever smaller parts, similar to the original. A fractal is generally "a rough or fragmented geometric shape that can be split into parts, each of which is (at least approximately) a reduced-size copy of the whole".[24] It is an irregular geometric object that is self-similar to its substructure at any level of refinement.

The general features of fractals [12] usually include fine structure at arbitrarily small scales, self-similarity (at least approximately or stochastically), and a simple and recursive definition. Fractal is too irregular to be easily described in traditional Euclidean geometric language, and it has a Hausdorff dimension which is greater than its topological dimension for most of the cases. (Some space-filling curves such as the Hilbert curve do not meet this requirement).

An amazing fact about fractals is the variety of their applications. Fractals can be used to describe a large number of highly irregular real-world objects. This includes the fractal landscape, signal and image compression, fractography and fracture mechanics, and the classification of histopathology slides in medicine. Among these, the widest and most mature application of fractal till now is in art creation, such as the generation of music and various art forms.

2.1.2 Fractal Art

Almost thirty years ago, research on the application of fractals in art and design field began to appear. It was first published in an article about the Mandelbrot Set in "Scientific American" in 1985. Starting with 2-dimensional fractals, such as the Mandelbrot Set, fractals have found applications in the fields of image and texture generation, plant growth simulation and landscape generation. Fractal art generally refers to two dimensional visual art at present, and is in many respects similar to photography — another art form which was greeted by skepticism upon its arrival. Though fractal images are not technically fractals, they have been welcomed into the fractal art world for the sharing of the same basic generating technique and environment. Fractal images typically are manifested as prints, bringing fractal artists into the company of painters, photographers, and printmakers.

Fractal Art is not computerized art, in which the computer does all the work. The work is executed on a computer, but only at the discretion of the artist. Fractal Art is not random in the sense of stochastic, or lacking any rules. Being based on mathematics, fractal rendering is determinate. Apply the same image generation steps, and the same result will be obtained. Slight changes in the process usually lead to slight changes in the result, making FA an activity which requires skill and experience, and is not a haphazard process of pushing buttons and turning knobs.

Making Fractal Art is a creative process. The final fractal image is created like photography or painting. The fractal artist begins with a blank "canvas" and creates an image, bringing together the same basic elements of color, composition, balance, etc., used by the traditional visual artist.

Fractal Art requires the intelligence of artists and their input. The Fractal Artist must direct the assembly of the calculation formulas, mappings, coloring schemes, palettes, and their requisite parameters. Each and every element can and will be tweaked, adjusted, aligned, and re-tweaked in order to find the right combination. The freedom to manipulate all these facets of a fractal image gives an artist the space to create, and this creation requires the understanding, intelligence and thoughtfulness of the artist [20].

2.2 Fractal Geometry

The observation by Mandelbrot of the existence of the "Geometry of Nature"[24] has led people to think in a new scientific way about the edges of clouds, the profiles of the tops of forests on the horizon, and the intricate moving arrangement of the feathers on the wings of a bird as it flies. Geometry is concerned with making our spatial intuitions objective. Classical geometry provides a first approximation to the structure of physical object. Fractal geometry is an extension of classical geometry. It can be used to make precise models of physical structures from ferns to galaxies. Fractal geometry is a new language, which can describe the shape of a cloud as precisely as an architect can describe a house.[3]

A few fundamentals need to be discussed regarding the mathematical background of Iterated Function System (IFS) theory. Barnsley [3] provided a complete reference on this topic. This chapter will review some of the most important results of IFS theory.

2.2.1 Metric Space

The spaces where fractals live are mathematically called metric spaces. Fractal geometry is concerned with the description, classification, analysis and observation of subsets of metric spaces *(X, d)*.

A metric space is an ordered pair *(X, d)* where X is a non-empty set and d is a metric on X, that is, a function

$$d: X \times X \rightarrow [0,\infty)$$

such that for any x, y and z in X

1. $d(x,y) = 0$ if and only if $x=y$ for all $x, y \in X$.

2. d(x,y) = d(y,x) for all $x, y \in X$.

3. d(x,y) $\leq$ d(x,z)+ d(z,y) for all $x, y, z \in X$.

A sequence $\{x_i\}_{i=1,2,...}$ in X converges to a point $x \in X$ in the metric space *(X, d)* if and only if the sequence $\{d(x_i, x)\}_{i=1,2,...}$ of numbers converges to zero.

A sequence $\{x_i\}_{i=1,2,...}$ in X is said to be bounded in the metric space *(X, d)* if there is a constant δ such that $d(x_i, x_j) \leq \delta$ for any integers *i, j*.

A sequence $\{x_i\}_{i=1,2,...}$ in X is said to be a Cauchy sequence in the metric space *(X, d)* if for any positive number ε > 0 there is a positive integer *k* such that $d(x_i, x_j) < \varepsilon$ for all integers $i, j > k$.

A metric space *(X, d)* is said to be ***complete*** if any Cauchy sequence $\{x_i\}_{i=1,2,...}$ in X converges to some point $x \in X$. Intuitively, this means no limit point in the space is missing. From now on, all metric spaces used in our applications will be assumed to be complete.

Let *(X, d)* be a complete metric space. Let $S \in X$. Then S is ***compact*** if and only if it is closed and totally bounded.

2.2.2 Contraction Mapping Theorem

A map $f : X \to X$ from the metric space (X, d) into itself is called a transformation. In general, in most applications, a transformation is expected to be bijective mapping. For any point x of X there is some unique point z of X to map into, $f(x) = z$, and there is also some unique point u of X to be mapped from, $f(u) = x$. Or equivalently,

f is invertible, i.e., there is another transform $f^{-1}: X \rightarrow X$, called the inverse of f, such that their composition is the identity:

$$f \circ f^{-1} = f \circ f^{-1} = identity$$

A transformation $f: X \rightarrow X$ on a metric space (X, d) is called contractive if there is a constant $0 \leq s < 1$ such that

$$d\left(f(x), f(y)\right) \leq s \cdot d(x, y)$$

for all $x, y \in X$. The number s is called a contractivity factor for the transformation f.

A point $a \in X$ is called a fixed point of the transformation f if $f(a) = a$.

The Contraction Mapping Theorem Let $f: X \rightarrow X$ be a contractive transformation on a complete metric space (X, d). Then the transformation f possesses exactly one fixed point $a \in X$. Moreover, for any $x \in X$, the sequence

$$x, f(x), f^2(x) = f(f(x)), \cdots, f^k(x) = f(f^{k-1}(x)), \cdots$$

converges to the fixed point a, i.e.,

$$\lim_{k \to \infty} f^k(x) = a$$

The fixed points of certain set are discovered as the ***attractor*** of fractals.

2.2.3 Transformations on metric space

A transformation f in the n-dimensional real space $\Re^n$ is a function that maps from $\Re^n$ to itself. The most interesting and useful transformations of real space $\Re^n$ are the affine transformations studied in linear algebra and the analytic transformations studied in complex analysis.

An affine transformation $f : \Re^n \to \Re^n$ is a transformation that can be written as

$$f(x) = Ax + b = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix} \bullet \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} + \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}$$

where $A = \left(a_{ij}\right)_{i,j=1}^{n}$ is an $n \times n$ matrix in $\Re^{n \times n}$, called the transformation matrix of f,

and $b = \left(b_j\right)_{j=1}^{n}$ is a vector in $\Re^n$, called the translation vector of f.

Given the norm $\|\bullet\|$ of the vector space $\Re^n$, the norm of an affine transformation f which is a transformation matrix A, is defined by the following formula:

$$\| f \| = \| A \|_p = \max_{x \in \Re^n, x \neq 0} \frac{\| Ax \|_p}{\| x \|_p}$$

It can be deduced that for the $p = 1, 2, \cdots, \infty$, their corresponding norms for the transformation matrix have the following intuitive formulas:

$\| A \|_1 = \max\limits_{1 \le j \le n} \sum\limits_{i=1}^{n} | a_{ij} |$,

$\| A \|_2 = \sqrt{\lambda_{\max}\left(A^T A\right)}$, $\lambda_{\max}$ is the largest eigenvalue of the matrix,

$\| A \|_\infty = \max\limits_{1 \le i \le n} \sum\limits_{j=1}^{n} | a_{ij} |$,

for any $n \times n$ matrix $A = \left(a_{ij}\right)_{i,j=1}^{n}$ in $\Re^{n \times n}$.

With the above definition, an affine transformation f with transformation matrix A is said to be contractive if $\| A \| < 1$. Furthermore, it is said to be contractive within the factor s, for some $0 \le s \le 1$, *if* $\| A \| < s$.

In the 3-dimensional case, $n = 3$, a transformation composes of scaling, rotating, shearing and translation.

a) Scaling

Uniform scaling is a linear transformation that enlarges or diminishes objects; the scaling factor is the same in all directions; it is also called a homothety. The result of uniform scaling is similar (in the geometric sense) to the original. Below is the transformation matrix for scaling in 3 dimensional space:

$$A_s = \begin{pmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & s \end{pmatrix}, s > 0$$

In general, scaling can be performed with a separate scale factor for each axis direction. Non-uniform or anisotropic scaling is obtained when at least one of the scaling factors is different from the others; a special case is directional scaling (in one direction). Non-uniform scaling changes the shape of the object; e.g. a square may change into a rectangle of a different shape. A general scaling in 3 dimensional Euclidean space can be represented by a scaling matrix as below:

$$A_s = \begin{pmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & s_3 \end{pmatrix}, s_i > 0, i = 1,2,3$$

b) Shearing

In mathematics, a shear mapping is a particular kind of linear mapping. Shearing is a transformation that effectively rotates one axis so that the axes are no longer perpendicular. A shear matrix is an elementary matrix that represents the addition of a multiple of one row or column to another. A simple shear matrix may be constructed by taking the identity matrix and replacing one of the zero elements with a non-zero value. i.e.

$$S = \begin{pmatrix} 1 & 0 & \lambda \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \lambda \neq 0$$

c) Rotation

A rotation is a rigid body movement of an object in a circular motion about a point fixed. This definition applies to rotations within both two and three dimensions (in a plane and in space, respectively.) In two-dimension, an object rotates around a center (or point) of rotation. In three-dimension, an object rotates about an axis. This follows from Euler's rotation theorem.

Rotations around the x, y and z axes are called principal rotations. Rotation around any axis can be performed by taking a rotation around the x axis, followed by a rotation around the y axis, and followed by a rotation around the z axis. Any spatial rotation can be decomposed into a combination of principal rotations. The three basic rotation matrices in three dimensions are:

$$R_x(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{pmatrix}$$

$$R_y(\theta) = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix}$$

$$R_z(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

These matrices represent rotations of an object relative to fixed coordinate axes, by an angle of θ.

d) Translation

In metric space, a translation moves every point by a fixed distance in the same direction. It can also be interpreted as the addition of a constant vector to every point, or as shifting the origin of the coordinate system. A translation f can be written as $f(x) = x + b$, where b is a fixed vector.

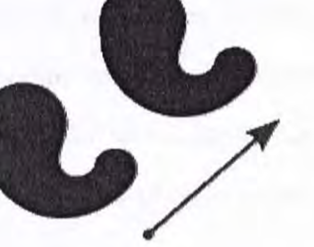


Figure 2.1: Translation

2.3 Construction of Fractals

Some common techniques for generating fractals include escape time method, Lindenmayer systems, stochastic synthesis and Iterated Function Systems.

Escape-time fractals (also known as orbits fractals) – These are defined by a formula or recurrence relation at each point in a space. Examples of this type are the Mandelbrot set, Julia set, the Burning Ship fractal, the Nova fractal and the Lyapunov fractal. The 2d vector fields that are generated by one or two iterations of escape-time formulae also give rise to a fractal form when points are passed through this field repeatedly.

Random fractals – Generated by stochastic rather than deterministic processes, for example, trajectories of the Brownian motion, Lévy flight, fractal landscapes and the Brownian tree. The latter yields so-called mass fractals, for example, diffusion-limited aggregation or reaction-limited aggregation clusters.

Strange attractors – Generated by iteration of a map or the solution of a system of initial-value differential equations that exhibit chaos.

Iterated function systems (IFS) – These systems require the use of a fixed geometric replacement rule. Cantor set, Sierpinski carpet, Sierpinski gasket, Peano curve, Koch snowflake, Harter-Highway dragon curve, T-Square, and Menger sponge are some examples of such fractals. Any set of linear maps or affine transformations associated with a set of probabilities determine an Iterated Function System. Each IFS has a unique attractor which is typically a fractal set. Specification of only a few maps can produce very complicated objects. Iterated function system is the tool we use to build fractals.

2.3.1 Iterated Function System (IFS)

An iterated function system consists of a complete metric space (X,d) together with a finite set of contraction mappings $w_k : X \to X$, with respective contractivity factors s_k, for k=1,2,...m. The contractivity factor s of IFS is defined to be the maximum of the contractivity factors of the transformations: $s = \max\{\| w_1 \|, \| w_2 \|, \cdots, \| w_m \|\}$.

Denote $\mathcal{H}(X)$ as a compact subsets space of the metric space (X,d). Given an IFS $W = \{w_1, w_2, \cdots, w_m\}$, define its associated transform in the space $\mathcal{H}(X)$, by

$$W(B) = w_1(B) \cup w_2(B) \cup \cdots \cup w_m(B),$$

for each $B \in \mathcal{H}(X)$.

Theorem 2.3.1: Let $W = \{w_1, w_2, \cdots, w_m\}$ be an iterated function system with contractivity factor s. Then its associated transform $W : \mathcal{H}(X) \to \mathcal{H}(X)$ is a contractive mapping in the space $\mathcal{H}(X)$ with the corresponding Hausdorff metric d with the same contractivity factor s. That is,

$$d(W(B), W(C)) \le s \cdot d(B, C),$$

for all $B, C \in \mathcal{H}(X)$. Its unique fixed point, $A \in \mathcal{H}(X)$ obeys

$$A = W(A) = \cup_{i=1}^{m} w_i(A)$$

and is given by $A = \lim_{i \to \infty} W^i(B)$ for any $B \in \mathcal{H}(X)$. The fixed point $A \in \mathcal{H}(X)$ described in the theorem is called the attractor of the IFS.

An example of constructing fractals by IFS is shown in below Figure 2.2. The diagram shows the construction of an IFS W, composed of two affine functions w$_1$ and w$_2$. The function transforms the outlined square into the shaded square respectively, and then the functions are represented by their effect on bi-unit square. Three iterations of the operator W are shown, and then the final image is the fixed point, the final fractal.

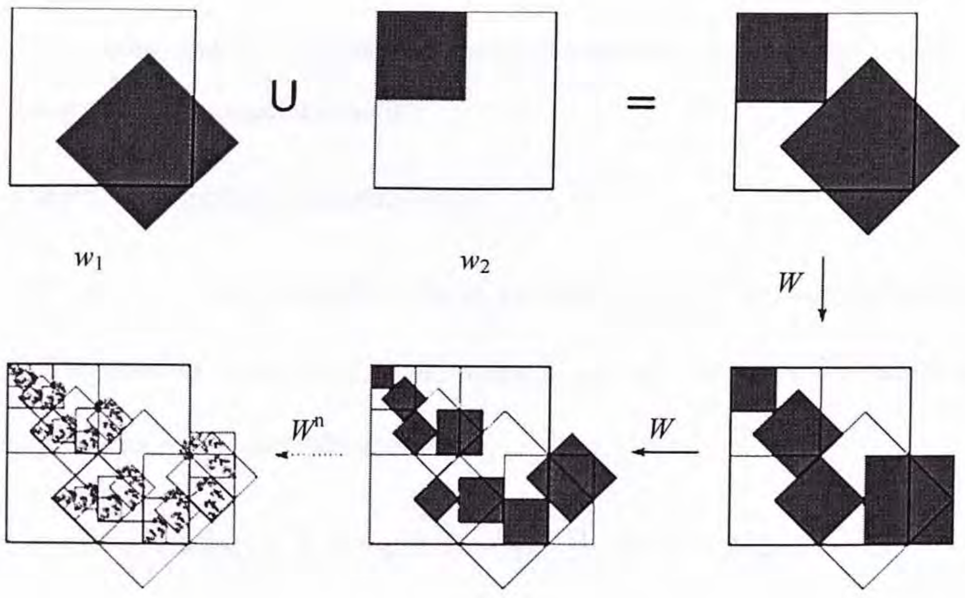


Figure 2.2: Procedure of constructing fractals by IFS [55]

2.3.2 IFS with Probabilistic Method

What will happen if one transformation is chosen much more frequently than the others in the random transformation selection procedure? To answer this question a new concept needs to be introduced first: IFS with probabilities.

An IFS $\{W : w_1, w_2, \cdots, w_m\}$ with probability $P = \{p_1, p_2, \cdots, p_m\}$ is an IFS with a positive number associated to each transformation; the sum of the probabilities is 1. That is,

$$p_i > 0 \; for \; all \; i = 1, 2, \ldots m, \; and \sum_{i=1}^{m} p_i = 1.$$

The Random Iteration Algorithm and Recurrent IFS are two fractal construction methods divided into this category. The Random Iteration Algorithm is originated in ergodic theory, and these probabilities play an important role in the visualization computation of the attractor of an IFS.

a) The Random Iteration Algorithm (RIA)

Let $\{W : w_1, w_2, \cdots, w_m\}$ be an IFS. Choose a compact set $A_0 \subset \mathfrak{R}^n$, with probability $p_i > 0$ assigned to w_i *for* $i = 1, 2, \ldots, m$, where $\sum_{i=1}^{m} p_i = 1$. Choose $x_0 \in X$ and then choose x_k recursively and independently,

$$x_k \in \{w_1(x_{k-1}), w_2(x_{k-1}), \ldots, w_m(x_{k-1})\} \quad for\ k = 1, 2, 3, \ldots,$$

where the probability of the event $x_k = w_i(x_{k-1})$ is p_i. Thus, construct a sequence $\{x_k : k = 0, 1, 2, 3, \ldots\} \subset X$.

To be efficient and to cover the space as quickly and evenly as possible, the best procedure is to set the value of each probability to be proportional to the volume of its corresponding transformation. The volume of a transformation is defined as the volume of the transformed unit cube, which is exactly the absolute value of the determinant of the deformation matrices. In conclusion, the default probabilities should be set to

$$p_i \approx \frac{|\det A_i|}{\sum_{i=1}^{m} |\det A_i|}$$

where $W_i(x) = A_i x + b_i$ *for all* $i = 1, 2, \ldots, m$.

Here the symbol $\approx$ means "approximately equal to". If for some i, $\det A_i = 0$, then p_i should be assigned a small positive number, such as 0.001.

b) Recurrent IFS (RIFS)

In RIA, the fractal construction uses a set of probabilities, one positive number for each transformation. What will the fractal look like if it is visualized with different probabilities other than the default one?

According to Theorem 2.3.1, it will converge to the same image or object, which is the attractor of the IFS. The images will appear identical after enough iterations. However, the procedure is much more different from the constructing procedure using RIA.

Barnsley, Elton and Hardin generalized the concept to that of Recurrent Iterated Function System. RIFS is an IFS with a set of probability sets, and one probability set is attached to each transformation.

Let $\{W : w_1, w_2, \cdots, w_m\}$ be an IFS. In a Recurrent IFS structure, there is a matrix $P = \{p_{ij}\}_{i,j=1,2,\ldots m}$ of probabilities with the following properties:

1. $\sum_{j=1}^{m} p_{ij} = 1 \; for \; all \; i, j = 1,2,\ldots m$

2. *for any* $i, j = 1,2,\ldots m$, there is some finite sequence

$$s_0, s_1, \ldots, s_k$$

such that $s_0 = i, s_k = j, \; and \prod_{b=0}^{k} p_{s_{b-1}s_b} > 0$. This is the irreducibility property of the RIFS.

The procedure of visualizing a fractal using RIFS is the same as using IFS with probabilities. Instead of following the same probability set all the time, an RIFS follows a probability set determined by the transformation in each iteration. That is, after applying the *i*th transformation, the probability set will be $\{p_{i1}, p_{i2}, \cdots, p_{im}\}$ for all $i = 1,2,\ldots$m.

2.3.3 IFS without Probabilities

IFS without probabilities is also called the deterministic algorithm. The Deterministic Algorithm is based on the idea of directly computing a sequence of sets $A_k = W(A_{k-1})$ starting from an initial set A_0; and the probabilities play no role in the Deterministic Algorithm.

The Deterministic Algorithm:

Let $\{W : w_1, w_2, \cdots, w_m\}$ be an IFS, and choose a compact set $A_0 \subset \Re^n$. Then compute successively according to

$$A_{k+1} = \bigcup_{i=1}^{m} w_i(A_k) \qquad \text{for } k = 1,2,\ldots$$

Thus construct a sequence $\{A_k : k = 0,1,2,3,\ldots\} \subset \mathcal{H}(X)$. Then by Theorem 2.3.1 the sequence $\{A_k\}$ converges to the attractor of the IFS in the Hausdorff metric.

2.4 Fractal Measurement and Aesthetics

2.4.1 Fractal Dimension

What is the size of a fractal? How to compare two fractals and decide how much similar they are? What measurements can be made to tell whether two fractals are metrically equivalent or not? A range of parameters can be employed to compare fractals, which are generally called fractal dimensions. These parameters quantify the subjective feeling of how densely a fractal occupies the metric space; it is also a measure of how complicated a self-similar object is. Fractal dimensions provide an objective means for comparing fractals.

Fractal dimensions are important because they can be defined in connection with real-world data, and they can be measured approximately by means of experiments. For example, one can measure the fractal dimension of the coastline of Great Britain, and its value is about 1.2. Fractal dimensions can be attached to clouds, trees, coastlines, feathers, networks of neurons in the body, dust in the air at a particular time instant, clothing, the distribution of frequencies of light reflected by a flower, the colors emitted by the sun, and the wrinkled surface of the sea during a storm. These numbers allow us to compare sets in the real world with those laboratory generated fractals, such as attractors of IFS.

Fractal dimension generally refers to any of the dimensions commonly used to characterize fractals, e.g., capacity dimension, correlation dimension, information dimension, Lyapunov dimension, Minkowski-Bouligand dimension. These parameters have a profound effect on the visual appearance of fractals.

2.4.2 Experimental Aesthetics

Our evaluation of 3d fractal aesthetics is based on the work of Clint Sprott [42][43]. This work proposed fractal dimension and Lyapunov exponent as a measure of complexity of a fractal image, and examined its relationship to aesthetic perception. The work of Sprott reported in [43] suggested a preference peak at correlation dimension 1.51 ± 0.43 and average Lyapunov exponent -0.24 ± 0.15 bits per iteration for 2D iterated function systems by averaging the 76 images rated 5 on a scale of 1 to 5. It was also found that for a given dimension, the largest negative values of Lyapunov exponent correspond to cases in which the two exponents are equal, implying the same contraction in all directions and perfect self-similarity. The largest negative Lyapunov exponent and fractal dimension are bounded by a curve,

$$-FL < \log O / \log D$$

where F is the correlation dimension, L is the largest Lyapunov exponent, O is the number of mappings and D is the dimension of the system of equations. Sprott also proposed an effective criterion to select visually appealing fractals automatically,

$$\left[(2-F)/1.2\right]^2+\left[(2+L/\log O)/1.6\right]^2<1.$$

Another groundwork is the work of Wannarumon S. etc. [50][51]. They used correlation analysis and linear regression to study hundreds of cases and built aesthetics function for 2d fractal pattern.

$$\begin{aligned}
fac_1 &= 0.304\cdot F_1+0.234\cdot F_2-0.202\cdot F_3+0.317\cdot F_4+0.186\cdot F_5-0.065\cdot F_6+0.17\cdot F_7+0.060\cdot F_8\\
fac_2 &= 0.103\cdot F_1-0.010\cdot F_2+0.077\cdot F_3+0.19\cdot F_4+0.085\cdot F_5-0.425\cdot F_6+0.428\cdot F_7+0.448\cdot F_8\\
\hat{S}_a &= -38.8442+39.0150\cdot e^{0.0287\cdot fac_1}-0.3917\cdot fac_2-0.6526\cdot fac_2^2+0.7553\cdot fac_2^3 \qquad (2.1)
\end{aligned}$$

where $F_1 \sim F_8$ are normalized capacity dimension, correlation dimension, largest Lyapunov exponent, image complexity, golden ratio, mirror symmetry, rotational symmetry, logarithmic spiral symmetry respectively. fac_1 measures aesthetics in terms of compactness, connectivity, and complexity of art forms that includes mirror symmetry. fac_2 represents aesthetics as rotational symmetry and logarithmic spiral symmetry. $\hat{S}_a$ is the mathematical aesthetics of fractal.

The visual attraction of fractals, including compactness, connectivity, regularity and symmetry is evaluated by a fitness value in this research and the fitness value will be further employed to sort the generated fractals and select parents in an evolutionary process.

3. OVERVIEW OF EVOLUTIONARY DESIGN

The use of evolutionary computation to generate designs has taken place in many different aspects over the last twenty years. Designers have optimized selected parts of their designs using evolution, artists used evolution to generate aesthetically pleasing forms, and computer scientists adopted evolutionary algorithm for simulating artificial life. Evolutionary Design is a general category that includes several research directions, such as evolutionary design optimization, creative evolutionary design, evolutionary art and evolutionary artificial life forms [5]. (See Figure 3.1)

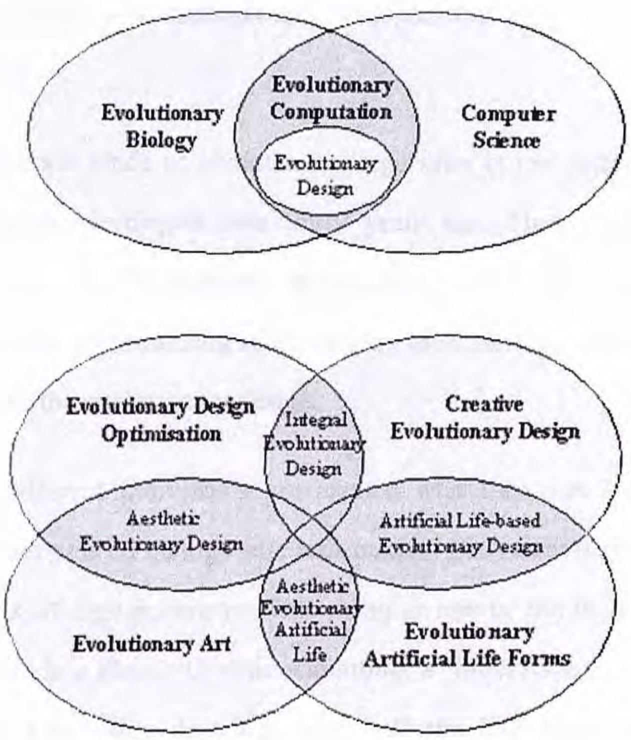
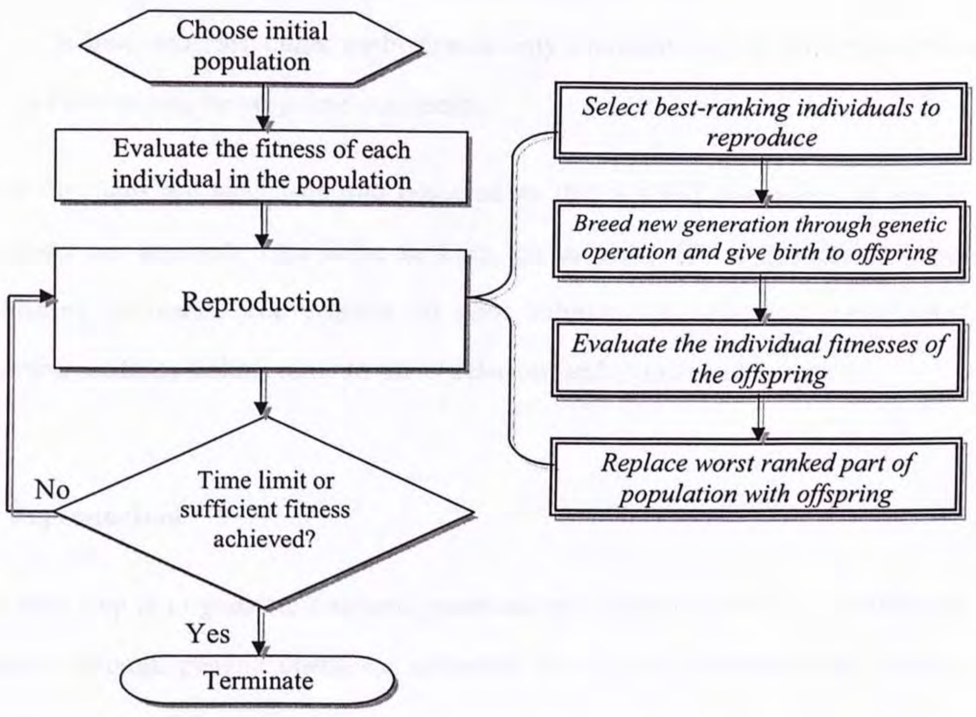

Evolutionary Biology
Evolutionary Computation
Evolutionary Design
Computer Science
Evolutionary Design Optimisation
Integral Evolutionary Design
Creative Evolutionary Design
Aesthetic Evolutionary Design
Artificial Life-based Evolutionary Design
Aesthetic Evolutionary Artificial Life
Evolutionary Art
Evolutionary Artificial Life Forms

Figure 3.1: The root of Evolutionary Design and Aspects of Evolutionary Design by Computers[5]

Creative evolutionary systems allow artists to develop stunning pieces of art, or allow musicians to create new sounds and new compositions. By using guided evolution, users are able to explore new ideas that emerge through the mechanisms of evolution. Other creative evolutionary systems take this approach one step further. Guidance is provided through automatic software controls that make judgment on evolving solutions without the need of human input. Designs are evolved from random blobs to fully functional forms. Novel circuits, ship hulls, architectural forms, even chemical structures are now routinely evolved by computers. This automatic generation of innovation by creative evolutionary systems allows designers to consider more solutions effectively. These systems allow us to sidestep limitations of "conventional wisdom" and "design fixation". Creative evolutionary systems can even suggest entirely new methods and principles that we can then exploit in our own designs. [6]

There are four main kinds of evolutionary algorithm in use today, three of which were independently developed over thirty years ago. These algorithms are: the genetic algorithm (GA), evolutionary programming (EP), and evolution strategies (ES) and the genetic programming (GP). Among these, GA is a simple and effective method applied to the evolutionary design.

In GA, many different individuals are created who then vie for the chance to reproduce and then pass on their genetic information. Each individual is described by a string or matrix of digit parameter. This string or matrix can be thought of as the individuals' DNA. In a classic Genetic Algorithm, a 'fitness value' is calculated for each individual. This value describes how well the individual's quality is. By selectively breeding individual with high fitness value, more successful art forms are reproduced. To produce a child individual from two parent individuals, some recombination of the two parent genes must occur. Once the genetic representation and the fitness function are defined, GA proceeds to initialize a population of solutions randomly, and then improves it through repetitive application of mutation, crossover, inversion and selection operators. Below is the procedure of a common genetic algorithm.



Choose initial population
Evaluate the fitness of each individual in the population
Reproduction
Time limit or sufficient fitness achieved?
No
Yes
Terminate
Select best-ranking individuals to reproduce
Breed new generation through genetic operation and give birth to offspring
Evaluate the individual fitnesses of the offspring
Replace worst ranked part of population with offspring

Figure 3.2: Flowchart of a typical genetic algorithm

3.1 Initialization

Initially, solutions are randomly generated to form an initial population. The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions. Traditionally, the population is generated randomly, covering the entire range of possible solutions. Occasionally, the solutions may be "seeded" in areas where optimal solutions are likely to be found.

3.2 Selection

During each successive generation, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected. Some selection methods rank the fitness of each solution to select the best solutions. Other methods rank only a random sample of the population, as this process may be very time-consuming.

Most functions are stochastic and designed so that a small proportion of less fit solutions are selected. This helps to keep the diversity of the population large, preventing premature convergence on poor solutions. Popular and well-studied selection methods include roulette wheel selection and tournament selection.

3.3 Reproduction

The next step is to generate a second generation population of solutions from those selected through genetic operators: crossover (also called recombination), and/or mutation.

For each new solution to be produced, a pair of "parent" solutions is selected for breeding from the pool selected previously. By producing a child solution using the methods of crossover and mutation, a new solution is created which typically shares many of the characteristics of its parents. New parents are selected to produce new child, and the process continues until a new population of solutions of the desired size is generated.

These processes ultimately result in a generation of chromosomes that is different from the initial generation. In general, the average fitness of the population will be increased by this procedure, since only the best organisms from the first generation are selected for breeding, along with a small proportion of less fit solutions.

3.3.1 Crossover Operation

Crossover varies changes the arrangement of chromosomes from one generation to the next. It is analogous to biological reproduction and crossover, upon which genetic algorithms are based.

a) One-point crossover

A single crossover point on both parents' chromosomes is selected. All data beyond that point in either organism string is swapped between the two parent organisms. The resulting organisms are the children:

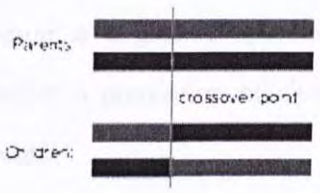


Figure 3.3: One-point crossover [44]

b) Two-point crossover

Two-point crossover calls for two points to be selected on the parent organism strings. Everything between the two points is swapped between the parent organisms, giving two child organisms:

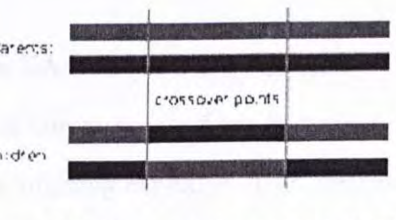


Figure 3.4: Two-point crossover [44]

c) "Cut and splice" crossover

Another crossover variant, the "cut and splice" approach, results in a change in length of the children strings. The reason for this difference is that each parent string has a separate choice of crossover point.

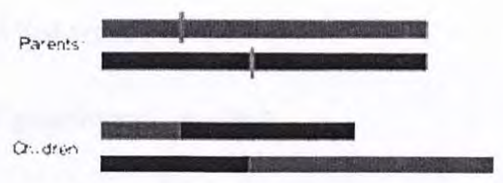


Figure 3.5: "Cut and splice" crossover [44]

3.3.2 Mutation Operation

In genetic algorithms, mutation is a genetic operator used to maintain genetic diversity from one generation of a population of chromosomes to the next. It is analogous to biological mutation.

The classic example of a mutation operator involves a probability that an arbitrary bit in a genetic sequence will be changed from its original state. A bit, can also be called allele, is one member of a pair or series of different forms of a gene, and are usually coding sequences. In other words, alleles are members of a gene that produce different traits in a gene's characteristics. A common method of implementing the mutation operator involves generating a random variable for each bit in a sequence. This random variable tells whether or not a particular bit will be modified.

The purpose of mutation in GA is to allow the algorithm to avoid local minimum by preventing the population of chromosomes from becoming too similar to each other, thus slowing down or even stopping evolution. This reasoning also explains the fact that most GA systems avoid only considering the fittest of the population in producing the next generation but rather a random (or semi-random) selection with a weighting toward those that are fitter.

3.4 Termination

This process for producing new generation is repeated until a termination condition is reached. Common terminating conditions are:

a) A solution is found that satisfies the criteria;

b) A fixed number of generations is reached;

c) The allocated budget (computation time) has been reached;

d) The solution with the best fitness is reached;

e) Manual inspection;

f) Combinations of the above.

4. EVOLUTIONARY 3D FRACTAL MODELING

This chapter gives a detailed description of the methodology we used in modeling the three dimensional fractals, such as how to construct fractals in three dimensional space using evolutionary algorithm. Finally, the transforming property and user fine-tuning are discussed.

4.1 Fractal Construction

Iterated Function System (IFS) is an effective way to model fractals and is widely used. It provides a very compact representation, efficient computation, and a very small amount of user specification can obtain a very large class of diverse fractals.

IFS is represented by a set of affine transformations, which can be any combination of scaling, rotation, shearing and translation of point sets. An IFS is defined by a set of contraction mappings w_k in complete metric space (X,d), where $w_k : X \to X$ with respective contraction factors s_k, $|s_k| \le 1$ *for* $k = 1,2,...m$, and k is the index for each affine map. An affine transformation of a point set in the Euclidean three dimensional space can be written in matrix form:

$$w_k(x,y,z) = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} d_1 \\ d_2 \\ d_3 \end{pmatrix} = A_k X + D_k \qquad (4.1)$$

where the elements a_{ij} control scaling, rotation and shearing, and d_i control translation.

4.1.1 Self-similar Condition of Fractal

The FT IFS (Fractal Transformation IFS) under the self-similar condition of fractal is formulated, which possesses straightforward geometric meaning. This provides intuitive user control on the shape of the fractal by tuning the parameters in each affine map.

Let $\Re^{n\times n}$ denote the set of $n \times n$ matrices and let A be a finite family of the expanding matrices $A_k \in \Re^{n\times n}, k = 1,2,...m$, m is the number of the matrices in the family (i.e. A_k, with all the module of its eigenvalues > 1, is an expanding matrix). Let $D=\{\overrightarrow{d_1},\overrightarrow{d_2},...\overrightarrow{d_m}\}\subset\Re^n$ be a set of n dimensional vector with real number component.

We define the affine maps $w_k(X)=A_k^{-1}\left(X+\overrightarrow{d_k}\right)$, and call $\{w_k(X)\}_{k=1}^{m}$ a self-affine iterated function system. If an attractor $T=T(A,D)$ satisfies $T=\bigcup_{k=1}^{m}w_k(T)$, then T is a self-affine set. For the special case that all A_k equal to A, the self-affine set T always exists under the expanding condition. If all the A_k are similar matrices, T is a self-similar set. In general, T or its boundary ∂T (if T has non-void interior) are fractal sets [17].

4.1.2 Fractal Transformation (FT) IFS Formulation

To adapt fractal modeling to three dimensional space, IFS is modified in this research and a FT (Fractal Transformation) IFS formula is put forward.

In the proposed method, a combination of rotation, translation and scaling in three dimensional Euclidean space is adopted to build fractals. Rotations about each axis are often used while transforming an object to a desired posture, position or coordinate systems. The proposed approach can also be viewed as a kind of IFS.

In the following discussion, the angle of rotation is specified with a right hand coordinate system. The rotation about the z axis will be referred to as *roll*, rotation about the y axis as *yaw*, and rotation about the x axis as *pitch* [49]. A rotation will be considered positive if it is clockwise when looking down the axis towards the origin. The proposed IFS equation is as follow:

$$\begin{cases} w_k(x,y,z) = A_k \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} d_1 \\ d_2 \\ d_3 \end{pmatrix} \qquad (4.2) \\ A_k = \rho \begin{pmatrix} \cos(\alpha_r)\cos(\alpha_y)+\sin(\alpha_r)\sin(\alpha_p)\sin(\alpha_y) & \sin(\alpha_r)\cos(\alpha_p) & -\cos(\alpha_r)\sin(\alpha_y)+\sin(\alpha_r)\sin(\alpha_p)\cos(\alpha_y) \\ -\sin(\alpha_r)\cos(\alpha_y)+\cos(\alpha_r)\sin(\alpha_p)\sin(\alpha_y) & \cos(\alpha_r)\cos(\alpha_p) & \sin(\alpha_r)\sin(\alpha_y)+\cos(\alpha_r)\sin(\alpha_p)\cos(\alpha_y) \\ \cos(\alpha_p)\sin(\alpha_y) & -\sin(\alpha_p) & \cos(\alpha_p)\cos(\alpha_y) \end{pmatrix} \end{cases}$$

Where ρ is the amplification factor, which ranges from 0 to 1; α_p is pitch, rotation angle about x axis; α_y represents yaw, rotation angle about y axis; α_r is roll, rotation angle about z axis. α_p, α_y and α_r range from $-\pi$ to π. See Figure 4.1, x is rotated to x'. This formula has a geometric meaning that A_k is a rotation in three dimensional space. The order of rotations applied here is to rotate about the y axis first (yaw), then the x axis (pitch), and then the z axis (roll).

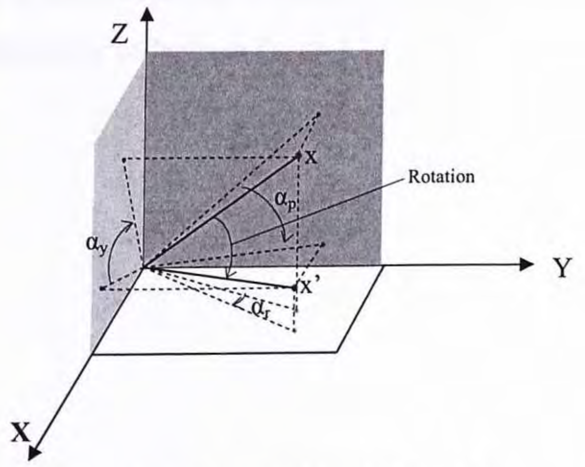


Figure 4.1: Rotation Decomposition in Eqn. (4.2)

According to Eqn. (4.2), it is clear that A_k^{-1} is the expanding orthonormal matrices, and all the A_k are similar matrices, so the proposed formula satisfies the self-similar condition described in sec. 4.1.1 and the self-similar set always exists. And according to contraction mapping theorem, the fractal construction is repeatable.

Compared to the original IFS, the FT IFS formula provides a more straightforward geometric meaning, which is the foundation for intuitive user interaction, and the productivity of good fractals is higher while using the FT IFS formula to build art fractals.

4.1.3 IFS Genotype and Phenotype Expression

Genotype is the "internally coded, inheritable information" describing an individual [39]. Genotype is the genetic information describing fractals in this research. An IFS is encoded as a genotype in the form of an N by 7 matrix. Each row of the matrix is a gene, which is encoded by rotation angles, scaling factor and the offsets of the affine maps. M. Barnsley and S. Demko [2][3] and physicist J.C. Sprott [43] etc. applied two to three affine maps in IFS. The fractal phenotypes' chromosome consisting of two or three genes can be easily controlled, N takes a value of 2 or 3 in this research. The N by 7 chromosome matrix is shown in Figure 4.2.

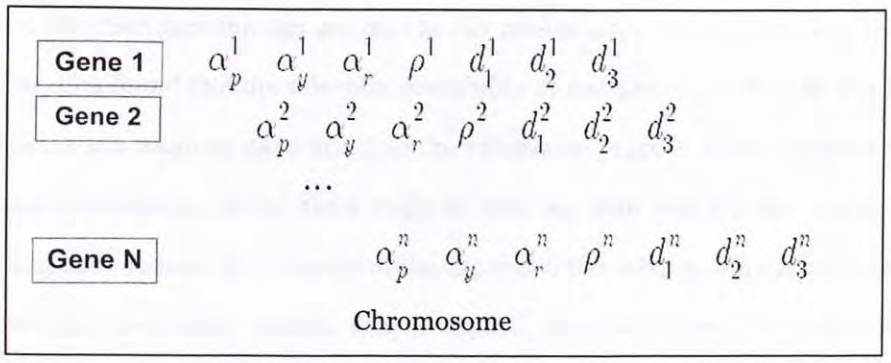

Gene 1
$\alpha_p^1 \quad \alpha_y^1 \quad \alpha_r^1 \quad \rho^1 \quad d_1^1 \quad d_2^1 \quad d_3^1$
Gene 2
$\alpha_p^2 \quad \alpha_y^2 \quad \alpha_r^2 \quad \rho^2 \quad d_1^2 \quad d_2^2 \quad d_3^2$
...
Gene N
$\alpha_p^n \quad \alpha_y^n \quad \alpha_r^n \quad \rho^n \quad d_1^n \quad d_2^n \quad d_3^n$
Chromosome

Figure 4.2: IFS Genotype

Phenotype is the "outward, physical manifestation" of the individual, which is the fractal individual generated with the genotype and development rule. Expression is the process of converting genotype to phenotype. The genotype is mapped to a phenotype through a probability selection process and a translation process. In this research, a revised Random Iteration Algorithm (RIA) is adopted for the probability selection. The affine map in the IFS formula is a gene in the genetic algorithm, and a fractal genotype corresponds to a chromosome. There are at least two genes in a chromosome. In this thesis two and three genes are used for a chromosome. The phenotype will be rendered as volume points or reconstructed as mesh surface later on.

Revised RIA

Barnsley put forward a Random Iteration Algorithm (RIA) [2], which is a popular selection probability assigning method for its fast convergence rate. The original RIA formula is as shown in Eqn. (4.3).

$$P_i = \frac{|\det A_i|}{\sum_{i=1}^{N} |A_i|}, and \sum_{i=1}^{N} P_i = 1 \, and \, P_i > 0 \qquad (4.3)$$

The RIA selection probabilities can lead to fast convergence while generating fractal. However, it is found that the selection probability of one gene according to this RIA rule may be too small to have effect in the expression process. If this happens to a two-gene chromosome fractal, there might be only one gene that actually works, and this will greatly reduce the diversity of the expressed phenotypes. On the other hand, if the random probability scheme [50] is applied, the convergence rate can not be guaranteed. Based on this consideration, a minimum probability $P_{\min}$ is introduced in this research.

$$P_{\min} = 1/(\alpha * N) \tag{4.4}$$

The influencing factor α, which is larger than 1, is used to adjust the diversity of the fractals. If α is large, the effect of a certain gene might be reduced and the phenotype of the generated fractal might be less diverse. This formula means there will be no selection probability lower than the minimum probability. If one probability P_i is below $P_{\min}$, it will be added to $P_{\min}$. Because the sum of all the selection probabilities is one, the gene with highest probability will be adjusted to $P_{\max}+P_i-P_{\min}$. This minimum probability scheme can reconcile the conflict between the diversity of the fractal pattern and the convergence rate.

4.2 Evolutionary Algorithm

Specifically, genetic algorithm is employed as the fractal evolutionary system in our research.

Before the individual fractals evolve to the next generation, selection of parents has been performed as the first stage in the process. The fitness value of fractal is calculated in this stage and parents are selected from the fractal chromosome library. Fitness is a reference used to decide the likelihood of survival and is used for selecting parents for generating successive generation. The selection process reflects the basic concept of survival and fittest in evolutionism and it is critical to the convergence of the evolution process. In the second stage, the fractal genotypes of new generation are reproduced based on the selected parents' chromosomes. Crossover and mutation, which are classic operations in genetic algorithm, are employed in this evolution process. Besides, an inferior elimination mechanism is introduced in case of the sudden occurrence of inferior fractal individual. The crossover operator applied here is single-point crossover, which recombines two chromosomes at a certain random position. The mutations operation creates variation for each allele with a certain occurrence probability. (Allele is an alternative unit in a gene found at the same place on a chromosome.) The stability of the process is related to the crossover occurrence probability and the mutation occurrence probability. If the occurrence probabilities are too high, the average quality of generated fractals will be unstable. The proposed evolutionary system will terminate when a pre-defined maximum number of generations or a certain number of individuals is reached.

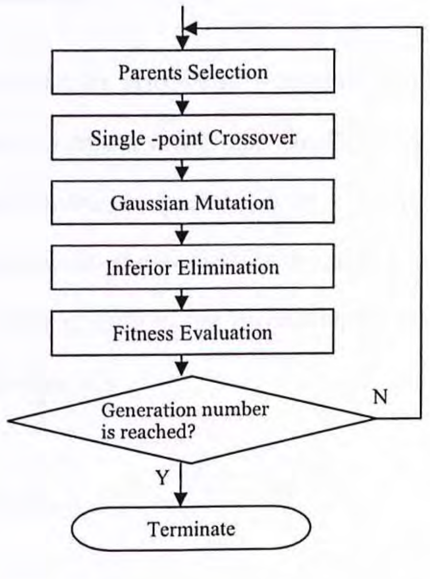


Figure 4.3: Flowchart of proposed Evolutionary System

4.2.1 Single-point Crossover

The crossover operator is to recombine portions of the chromosomes of fractal individuals, by which variation is generated to the given chromosome. The crossover position is created randomly. However, genes will not be exchanged wholly, which means the intervals between genes are excluded from crossover position (intervals, j=7 in the chromosome matrix O). Crossover probability P_c controls the occurrence of crossover operation.

$$O_1^{'} = \begin{bmatrix} C_1(1:i,1:j) & C_1(1:i,j:7) \\ C_2(i:N,1:j) & C_2(i:N,j:7) \end{bmatrix}$$
$$O_2^{'} = \begin{bmatrix} C_2(1:i,1:j) & C_2(1:i,j:7) \\ C_1(i:N,1:j) & C_1(i:N,j:7) \end{bmatrix} \tag{4.5}$$

where (i, j) is the crossover point, $o_1^{'}$ and $o_2^{'}$ are the chromosomes of the offspring, C_1 and C_2 are the chromosome matrix of parents, and C(1:i, 1:j) means submatrix of C from row 1 to row i and column 1 to column j.

4.2.2 Arithmetic Gaussian mutation

The mutation operation produces an arithmetic Gaussian random value for each allele in the gene pool. Mutation might bring uncontrollable changes to gene, and hence is strictly controlled by mutation probability P_m. If the mutations are less favorable, they will be reduced in the selection process and not pass to next generation; on the contrary, if the mutations are favorable, they may accumulate and result in adaptive evolutionary changes.

$$O = O' + M,\ M(i,j) = \{ \begin{matrix} k, & if\ prob \geq P_m \\ 0, & else \end{matrix} \tag{4.6}$$

where k is a Gaussian distribution random number with mean 0, variance 1 and standard deviation 1; *O* is the new offspring, and *O'* is the chromosome matrix after crossover operation.

4.2.3 Inferior Elimination

Due to the randomness of the evolution process, there might be sudden occurrence of quality decline for the generated fractals. Thus an inferior elimination mechanism is introduced in the evolutionary system. If the fitness of the chromosomes generated by the crossover and mutation operations is below a pre-defined cut-off score, the individual will not be passed to the parents' selection process and will be viewed as unqualified and eliminated. This mechanism can obviate inferior chromosomes and enhance the average quality of the generated fractal.

4.3 Interactive Fine-tuning using FT IFS

Design of fractal objects is made relatively simple and intuitive by the transforming property and the straightforward geometric meaning of the parameters in the revised FT IFS. After fractal is built by the evolutionary system automatically, users may further modify the fractal pattern to a desired shape.

Parameters provided by the formula can be tuned to modify the corresponding 3D fractal pattern. A fractal is a sequence of 3d points, and each successive point is obtained from transforming the previous 3d point. The parameters control the position of 3d point in each iteration, and thus affect the shape of fractals as a whole. As shown in Fig.4.4, x_k is transformed to x_{k+1} through rotation, scaling and translation. The rotation is a combination of the rotations about x, y, z axis. Parameters α_p, α_y and α_r are to expand or shrink the orbit radius about their respective axis. If α_p is large, the fractal object is farther away from the x axis far. ρ is the scaling factor in each iteration, and it changes the volume of the fractals. ρ should not be larger than 1, otherwise, the fractal will spread all over the space. The translation vector is the vector with exponent of d_1, d_2 and d_3, which can adjust the density and the span length along one coordinate axis. Besides, the selection probability for affine map can be changed to adjust the influence of one affine map in the phenotype mapping process. If a user considers one affine map is excellent, he can increase the effect of that map. An IFS system may have extremely high selection probability or comparably very low selection probability. When this occurs, those with high selection probability are dominant and those with low selection probability are called subsidiary affine maps. For example, when there are two affine maps, the selection probability of one is much higher than the other's, and then they are dominant affine map and subsidiary map respectively. On the contrary, if all the affine maps share relatively similar selection probabilities, there will be no dominant and subsidiary affine maps. In general, the dominant affine map usually can not take effect if the contribution of the subsidiary affine maps is too small. This means the selection probability of the subsidiary affine maps should not be zero.

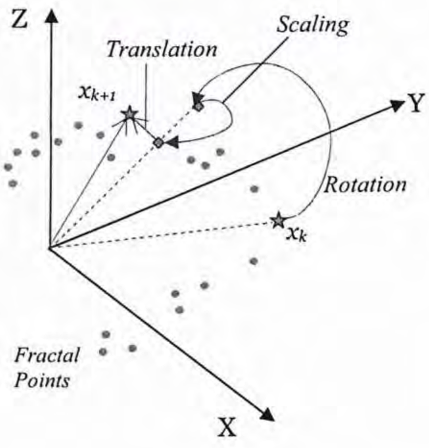


Figure 4.4: Transformation from x_k to x_{k+1} in an iteration

Through tuning these geometric parameters, users can modify the fractal pattern according to their own wills instead of complete random generation. As for the computing time, reproducing a fractal with changed parameters only needs a few seconds. An example is shown in Chapter 7.

4.4 Gaussian Fitness Function

After the fractals are built by the proposed FT IFS formula and are expressed to phenotypes, their fitness values need to be evaluated for further processes in the evolutionary system. The fitness of fractals reflects the quality of phenotype and generally represents their visual aesthetic appeal. They are used to rank the fractals. We formulated a Gaussian like fitness function by considering the mathematical quantities of fractal based on a large number of samples. The fitness value computed is then employed to select parents for the next generation in the evolutionary system. This fitness function will be presented in detail in the next chapter.

5. GAUSSIAN AESTHETIC FITNESS FUNCTION

Fitness describes the capability of an individual of certain genotype to reproduce, and usually is equal to the proportion of the individual's genes in all the genes of the next generation. If differences in individual genotypes affect fitness, then the frequencies of the genotypes will change over generations; the genotypes with higher fitness become more common. This process is called natural selection. As fitness measures the quantity of the copies of the genes of an individual in the next generation, it doesn't really matter how the genes arrive in the next generation. That is, for an individual it is equally "beneficial" to reproduce itself, or to help relatives with similar genes to reproduce, as long as similar amount of copies of individual's genes get passed on to the next generation.

An individual's fitness is manifested through its phenotype. As phenotype is affected by both genes and environment, the fitnesses of different individuals with the same genotype are not necessarily equal, but depend on the environment in which the individuals live, and the environment is the selection probability in this research. However, since the fitness of the genotype is an averaged quantity, it will reflect the reproductive outcomes of all individuals with that genotype.

Fractal fitness refers to the compactness, connectivity, regularity, symmetry and most importantly the aesthetic appeal. There are a number of different techniques for measuring these properties. In [42][43], a relationship between visual aesthetic judgments of fractal, their fractal dimensions and Lyaponov exponent has been proved and a Gaussian-format formula was used to evaluate the fitness function with two parameters.

Fractal dimension is a statistical quantity that gives an indication of how complete a fractal appears to fill a given space. The capacity dimension and correlation dimension are widely used in practice, partially due to their ease of implementation and fast computation.

The appealing of fractals, including compactness, connectivity, regularity and symmetry is evaluated by a fitness value in this research. This fitness value is used to rank the generated fractals so as to select parents in the evolutionary process. This quantitative measure of the aesthetics of fractal is formulated based on the considerations described in section 5.1. This formula will be used as the fitness function in the evolutionary algorithm to enforce the evolution of fractals. In the following discussions, we denote Capacity dimension as D_0, Correlation dimension as D_2, and Largest Lyapunov exponent as LLE.

5.1 Fitness Considerations

5.1.1 Capacity Dimension

One of the essential features of a fractal is that its Hausdorff dimension strictly exceeds its topological dimension [41]. In mathematics, capacity dimension, also named the Hausdorff–Besicovitch dimension or Hausdorff dimension, is an extended non-negative real number associated with a metric space. The capacity dimension generalizes the notion of the dimension of a real vector space. In particular, the capacity dimension of a single point is zero, the capacity dimension of a line is one, the capacity dimension of the plane is two, etc. There are however irregular sets that have noninteger capacity dimension.

The capacity dimension can be calculated using box-counting method [22], which is a way of determining the fractal dimension of a set S in a Euclidean space $\Re^n$ or more generally in a metric space (X,d). Assume the fractal lying on an evenly-spaced grid or putting the three dimensional fractal in a cube, and count how many boxes are required to cover the set. Making the boxes smaller gives more detail, which is the same as increasing the magnification. The box-counting dimension is calculated by considering how this number changes while making the grid finer. Suppose that N(ε) is the number of boxes of side length ε required to cover the set. Then the box-counting dimension of a fractal S is defined as:

$$D_0(S) = \lim_{\varepsilon \to \infty} \frac{\log(N(\varepsilon))}{\log(1/\varepsilon)}$$

As we investigate on three dimension fractals, the D_0 generated in our experiments ranges from 0 to 3, and it is normalized to [0, 1] by dividing D_0 by 3.

5.1.2 Correlation Dimension

The correlation dimension is a measure of the dimensionality of the space occupied by a set of random points. Compared with other fractal dimension, such as capacity dimension, D_2 also measures the contraction rate of the points that land on a fractal. D_2 can be calculated using the distances $s(i,j)$ between each pair of points X_i, X_j in the set of N number of points.

$$s(i,j) = |X_i - X_j|,$$

A correlation function, C(r), is then calculated using,

$$C(r) = \frac{1}{N^2} \times (number\ of\ pairs(i,j)\ with\ s(i,j) < r),$$

C(r) has been found to follow a power law: $C(r) = kr^{D_2}$. Therefore, D_2 can be derived with estimation techniques from the formula:

$$D_2(S) = \lim_{r \to \infty} \frac{\log(C(r))}{\log(r)},$$

C(r) can be written as

$$C(r) = \lim_{N\to\infty} \frac{1}{N^2} \sum_{j=1}^{N} \sum_{i=j+1}^{N} \theta\left(r - \left|X_i - X_j\right|\right),$$

where θ is the Heaviside step function described as,

$$\theta\left(r - \left|X_i - X_j\right|\right) = \begin{cases} 1 & r - \left|X_i - X_j\right| \geq 0 \\ 0 & r - \left|X_i - X_j\right| < 0 \end{cases}.$$

We use the Peter G. and Itamar P.'s correlation integral method [16], TSTool for nonlinear time series analysis [25] and linear regression to compute the correlation dimension. D_2 of 3D fractals is in the range of [0, 3], we normalize D_2 into the range of [0, 1]. The higher is D_2, the wider the fractal globally spread.

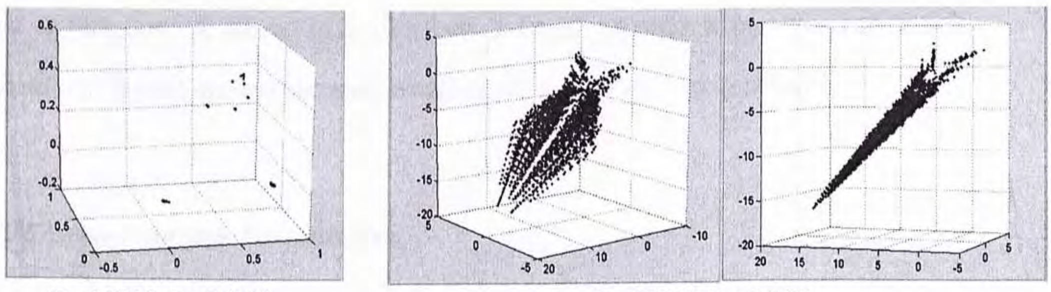


Figure 5.1: Comparison of fractal with different D_0 and D_2

5.1.3 Largest Lyapunov Exponent (LLE)

The Lyapunov exponent or Lyapunov characteristic exponent of a dynamic system is a quantity that characterizes the rate of separation of infinitesimally close trajectories. Quantitatively, the separation of two trajectories in phase space with initial separation $\delta\mathbf{Z}_0$ can be expressed as

$$\left|\delta Z(t)\right| \approx e^{\lambda t} \left|\delta Z_0\right|$$

where λ is the Lyapunov exponent.

Lyapunov exponent can be thought of as the rate at which information about the initial condition is lost. A negative value means that information is gained. The rate of separation can be different for different orientations of initial separation vector. Thus, there is a whole spectrum of Lyapunov exponents—the number of the exponents in a spectrum is equal to the number of dimensions of the phase space. It is common to just refer to the largest one, i.e. to the Largest Lyapunov exponent (LLE), because it is the one that dominates in the subsequent iterations and determines the predictability of a dynamic system. A positive LLE is usually taken as an indication that the system is chaotic.

We use Eckmann J. P. and Ruelle D.'s ergodic theory [11] and LET tool [1] to calculate the largest Lyapunov exponent, which ranges from -1.4123 to -0.0033 in this experiment. A lower LLE indicates a larger separation between two nearby points; the fractal pattern becomes more disconnected and less compact [51].

5.2 Fitness Function Formulation

In section 2.4.2, linear log equation is proposed to calculate the fitness and aesthetic value of 2D fractals [50]. However, experiment showed that the correlativity of fitness value and fractal characteristic parameters using linear equation is quite low. Sprott [43] proposed to use a quadratic sum equation to select nice fractal on 2D space and proved its effectiveness. Here we propose to employ Gaussian function to build the fitness function due to its characteristic symmetric bell shape curve that quickly falls off towards infinity, which can enhance the correlativity under the circumstance.

The fitness equation in evolutionary process classifies fractal patterns as qualified or unqualified based on an acceptance criteria. First, fitness score are rated. There are five items in the rating list, including overall appealing, compactness, connectivity, inerratic, and symmetry, and each of them takes 20%, and the final fitness score is normalized to [0, 1]. Secondly, we statistically analyze relationships between variables D_0, D_2, LLE and fitness. It is found that capacity dimension, correlation dimension, and the largest Lyapunov exponent have influence on fractal fitness. Capacity dimension indicates how effective a fractal fills up the space and its density. Correlation dimension suggests the fractal's dimension and its contraction rate. The Largest Lyapunov exponent estimates the separations of points in a fractal. The fitness of a fractal S can be formulized as:

$$fit(S) = a \cdot e^{-(\frac{(D_0 - d_0)^2}{2\sigma_1^2} + \frac{(D_2 - d_2)^2}{2\sigma_2^2} + \frac{(LLE - lle)^2}{2\sigma_3^2})} + b$$

where d_0 is the mean for variable D_0, d_2 is the mean for variable D_2, lle is the mean for LLE, σ_1, σ_2 and σ_3 are corresponding variances in Gaussian function, a and b are the linear coefficients.

According to Jacob [18], a multiple regression/correlation analysis, with the significance criterion α equal to 0.05 and small population effect size, requires a sample size of 541. Since there are three variables in the regression, the total sample size should be 1641. Before rating the samples, interrater reliability was built up, and the reliability is 80%. The range of D_0 for all sample cases is from 0.271 to 0.849; D_2 is from 0.035 to 0.980; and LLE is from -1.4123 to -0.0033. For the 29 cases that were rated best out of 1641 samples (fitness score equal or above 0.9), the average capacity dimension is $D_0 = 0.334 \pm 0.038$, the average correlation dimension is $D_2 = 0.587 \pm 0.326$, the average Largest Lyapunov exponent is $LLE = -0.057 \pm 0.046$. So $d_0 = 0.334$, $d_2 = 0.587$, $lle = -0.057$. Take $\ln(fit + 0.5)$ as dependent variable, $(D_0 - 0.334)^2$, $(D_2 - 0.587)^2$, and $(LLE + 0.057)^2$ as variables, correlation analysis and linear regression is used to analyze all sample cases for studying their relationships. The correlativity of fitness function built by Gaussian equation in this research is 0.71. The fitness function after normalization is educed as follow:

$$fit(S) = 1.493 \times e^{-\left(\frac{(D_0-0.334)^2}{2\times0.840^2}+\frac{(D_2-0.587)^2}{2\times0.698^2}+\frac{(LLE+0.057)^2}{2\times1.092^2}\right)} - 0.5 \qquad (5.1)$$

$fit(S)$ is the fitness value of fractal S, which lies in the range [0, 1].

5.3 Results and Discussion on Fitness Function

Fractals with different fitness value are shown in Figure 5.2. The number of voxels used for visualization the examples is 5000.

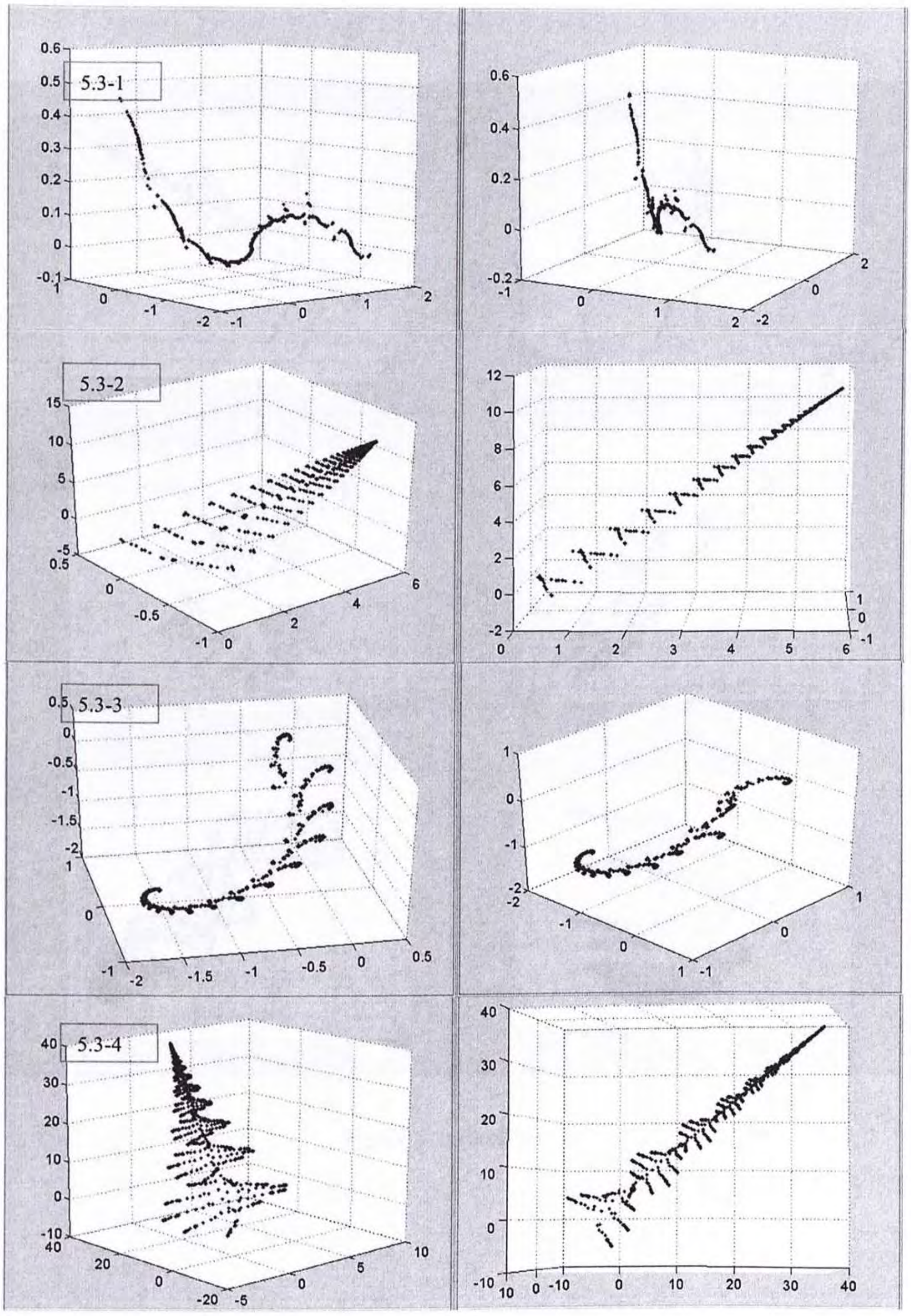


Figure 5.2: Fractals with different fitness value, the view is rotated clockwisely from left to right

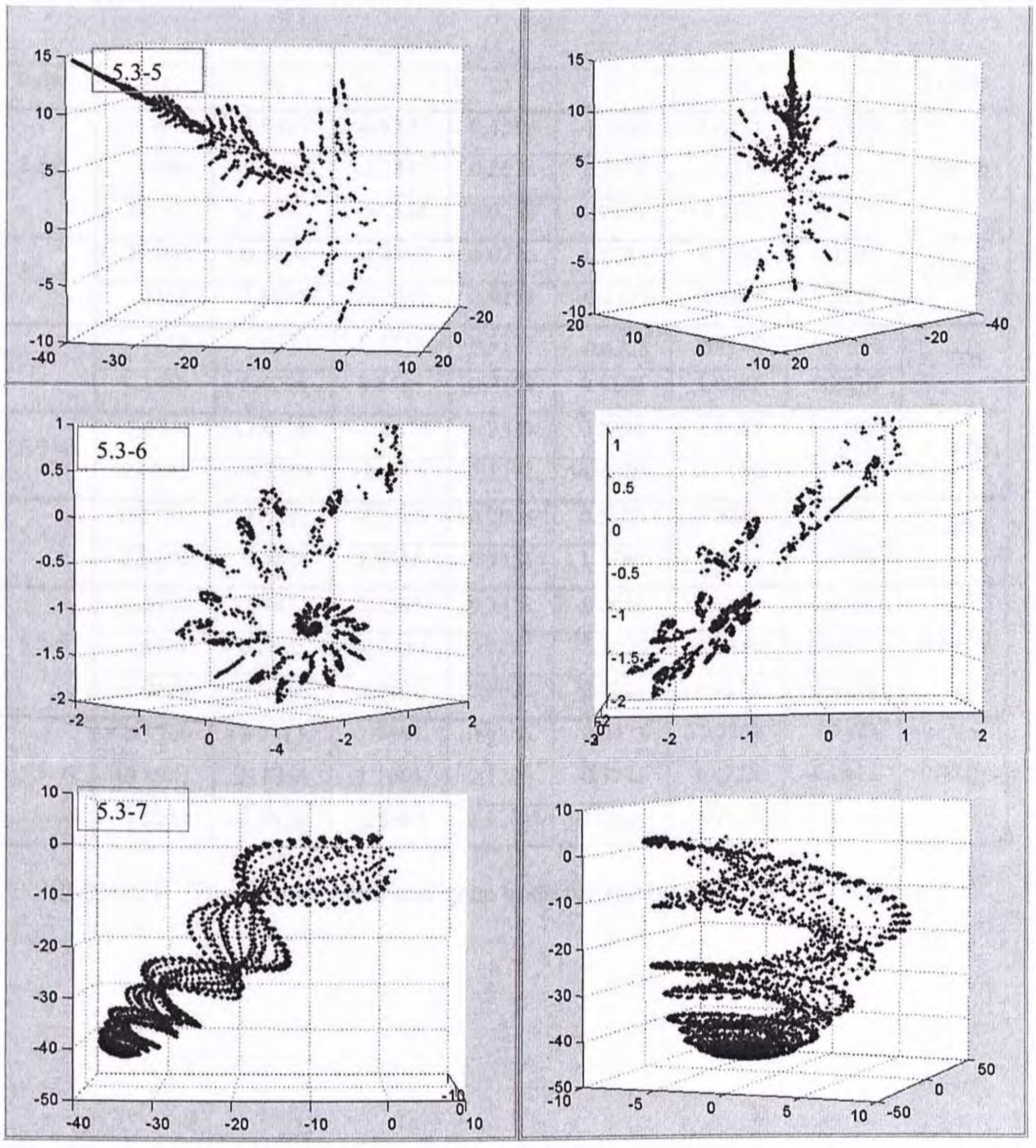


Figure 5.2: (cont'd)

The corresponding fitness value and gene code for these fractal pattern is as below in Table 5.1.

Index	α_p	α_y	α_r	ρ	d_1	d_2	d_3	Fitness
5.3-1	-2.2409	-1.0422	-0.9395	0.8569	-1.1245	1.6351	-0.1989	0.6688
	1.6690	-0.1722	-2.3885	0.2574	1.7357	-1.2559	0.3075	
	2.4887	-2.0523	-0.0114	0.0255	1.9375	-0.2135	-0.5723	
5.3-2	-2.6096	0.2659	-0.4418	0.0726	1.7221	0.5651	0.2201	0.6734
	0.0365	0.8885	-3.0947	0.9273	-0.4123	0.7399	1.3128	
5.3-3	1.9545	-1.0370	1.4752	0.3177	-0.0228	0.8128	-1.0046	0.7229
	1.2910	0.4296	0.8534	0.7758	0.1106	1.0091	0.2830	
5.3-4	-1.8816	-1.4478	-1.8274	0.9539	0.7358	1.4443	0.6863	0.7490
	2.6635	-0.2329	-2.5700	0.1312	-0.6409	-1.1590	0.7304	
5.3-5	0.2535	-2.5441	-1.3460	0.2008	0.5809	-0.9381	0.3760	0.7499
	2.9436	-0.4846	2.0856	0.9155	1.7786	-0.9167	0.9098	
5.3-6	-2.2188	-1.4472	0.5363	0.3458	-0.2656	0.9863	0.2341	0.8380
	1.5405	0.8805	-0.0215	0.9267	-1.1878	-0.5186	0.0215	
	-0.0130	-1.2466	2.4961	0.2892	-2.2023	0.3274	-1.0039	
5.3-7	-3.1200	3.0313	2.9841	0.9845	-0.9471	-1.0559	-1.2173	0.9719
	-0.8081	-0.6351	3.1063	0.1709	-0.3744	1.4725	-0.0412	
	-2.8234	1.8679	-2.5867	0.1784	-1.1859	0.0557	-1.1283	

Table 5.1: The fitness value and gene code for fractal examples in Figure 5.2

6. EXPERIMENT RESULTS and DISCUSSION

6.1 Experiment of Evolutionary Generation

A chromosome library, which contains a hundred of fractal chromosomes, is built for the initial selection of parents. The library is dynamic, and each time when new offspring are reproduced the library will be refreshed and enlarged automatically. According to Eqn. (4.4), the influencing factor α is set to 4 and the minimum mapping probability P_{min}, which is applied in the mapping process from genotype to phenotype, is set to 0.125 when N=2, and P_{min} is set to 0.083 when N=3. To control the stability of the evolution, the mutation probability P_m is assigned to 0.5, and the crossover probability P_c is assigned to 0.75. The cut-off score used in inferior elimination is 0.6. The termination criterion of the proposed evolutionary system is to reach a pre-defined maximum number of generations or a certain number of individuals. In the experimental system, the termination criterion is set to 15 generations. The iteration number used in fractal phenotype expression is 5000. The experiments are conducted on a PC with Intel Pentium 4, dual CPU 3.40 GHz, 2GB RAM. The average running time of reproducing a qualified fractal is 7.15 sec.

From Figure 6.1 we can see that, the average fitness value of the offsprings is all over 0.8, which means the quality of the fractal generated by employing the proposed evolutionary system is acceptable and stable among different generations. We systematically examined all the fractals generated in the evolution process and found that they are nearly all visually interesting.

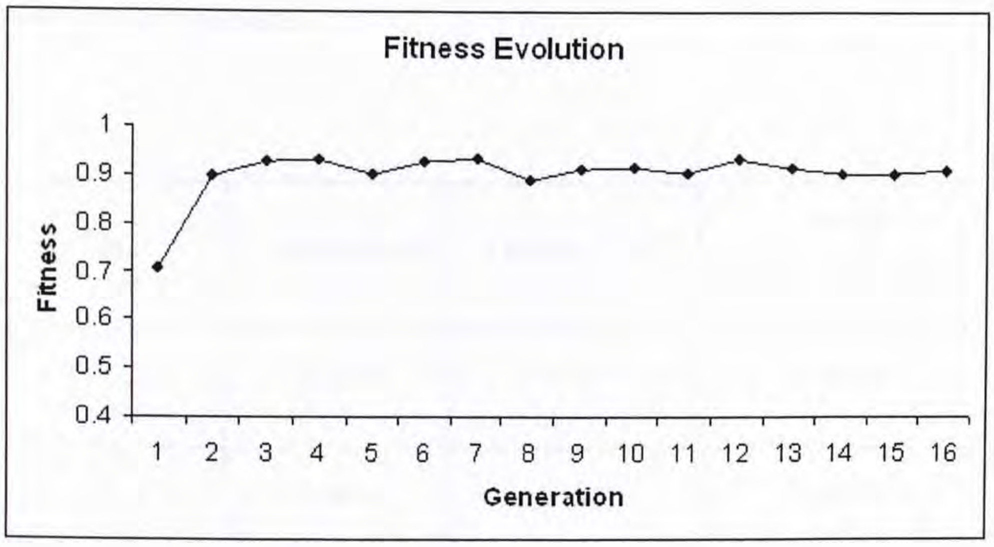


Figure 6.1: Average fitness value of fractals generated in the evolution process

6.2 Comparison on Different Methods

We conducted a comparison experiment to explore the effectiveness of fractal generation using different formulas and evolutionary systems. Three different methods are compared. In the first case, Eqn. (4.1) under contraction condition (i.e. A is random orthonormal matrix with |detA| less than 1) is employed to build a hundred of fractals. In the second case, Eqn. (4.2) is used to build a hundred fractals. In the third case, fractals are built by Eqn. (4.2) combined with evolutionary algorithm. There are 15 generations totally and each generation has a population size of 10. Considering to the stochastic characteristic of the evolutionary system, the experiments were conducted for ten times, and all the values shown in the below table are the average values from the ten experiments.

As we can see from Table 6.1, average fitness and excellent rate are increasing from case 1 to case 2 and from case 2 to case 3, and the average running time is also increased. The data demonstrates that using Eqn. (4.2) with evolutionary system can effectively enhance the quality of the generated fractals and the productivity of the fractals generation process.

Case	Average Fitness	Excellent Rate*	Average Running Time (sec.)
1	0.704	0.320	332.48
2	0.840	0.542	441.42
3	0.912	0.868	1072.35

Table 6.1: Some primary parameters for the three cases

*: Excellent rate means the percentage of fractals with fitness larger than 0.85.

7. 3D FRACTALS RENDERING and APPLICATION

The property of the fractal transform is the foundation for intuitive user interaction. It is illustrated with some tests in this chapter. Moreover, the visualization and rendering influence how the three dimensional fractals can be applied to a great extent. Based on the visualization and rendering method applied, various applications of three dimensional fractals are discussed.

7.1 Transforming Property and User Modification

7.1.1 Test on Transforming Property

According to Eqn. (4.2), the shape of the fractals transforms smoothly while the variables are changed. This is shown in Figure 7.1. Figure 7.1(a) shows the original appearance of fractal 7.1.1-1. Figure 7.1 (b, c, d) shows the result of changing the pitch, yaw, and roll by adding a value of $\pm\pi/30$ in each step. Figure 7.1(e) illustrates the result of changing d_1 by a value of 0.1. Figure 7.1(f) shows the effects of reducing ρ by 5% in each step. There are 10000 voxels used for the rendering.

Parameters α_p, α_y and α_r control the rotation angle about each axis, which can be used to enlarge or reduce the orbit radius about an axis, and the range of α is $[0, \pi]$. As shown in Figure 7.1 (b), if α_p is large, the fractal object moves away from the x axis. ρ influences the scale of the fractals, which can be used to decide the size of a fractal. However, ρ should lie in range [0, 1], otherwise, the fractal will not converge. d_1, d_2 and d_3 are offsets in the x, y, z axes applied in each iteration, and they can be used to adjust the density and the span length in each coordinate direction. If d_1 is large, the fractal is loose along the x axis.

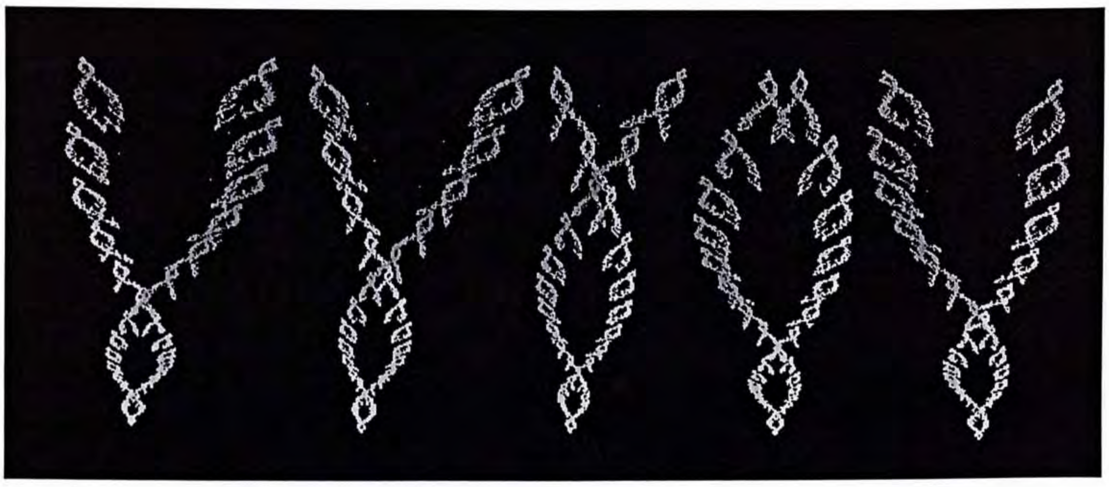


Figure 7.1(a): Multi-views of the 3D fractal 7.1.1-1, rotated clockwise from left to right

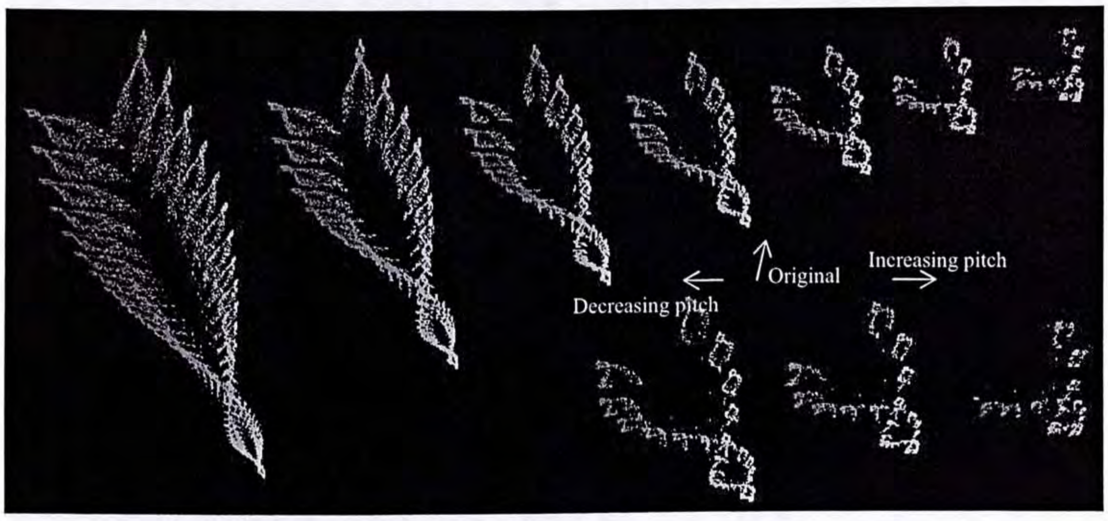


Figure 7.1(b): Effect of changing pitch, the bottom-right corner is the enlarged view of the rightmost three fractals

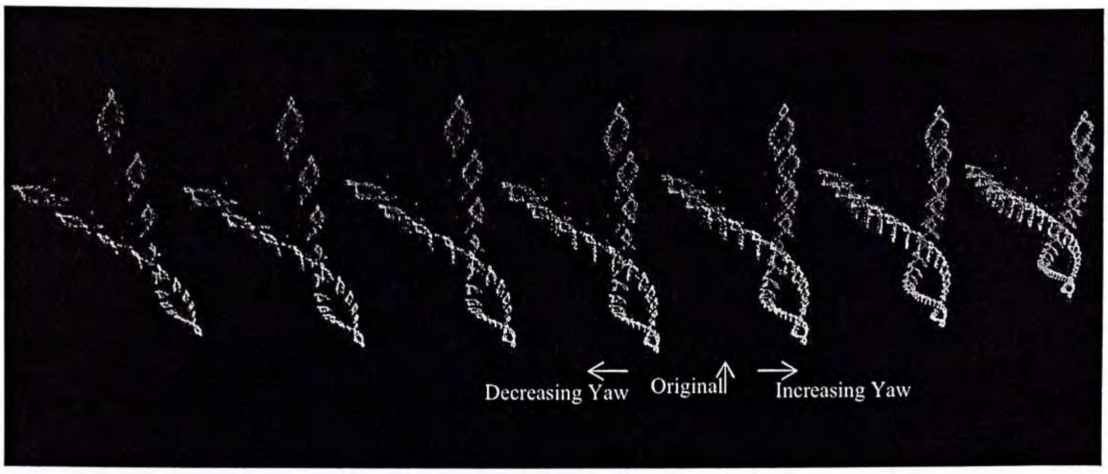


Figure 7.1(c): Effect of changing yaw.

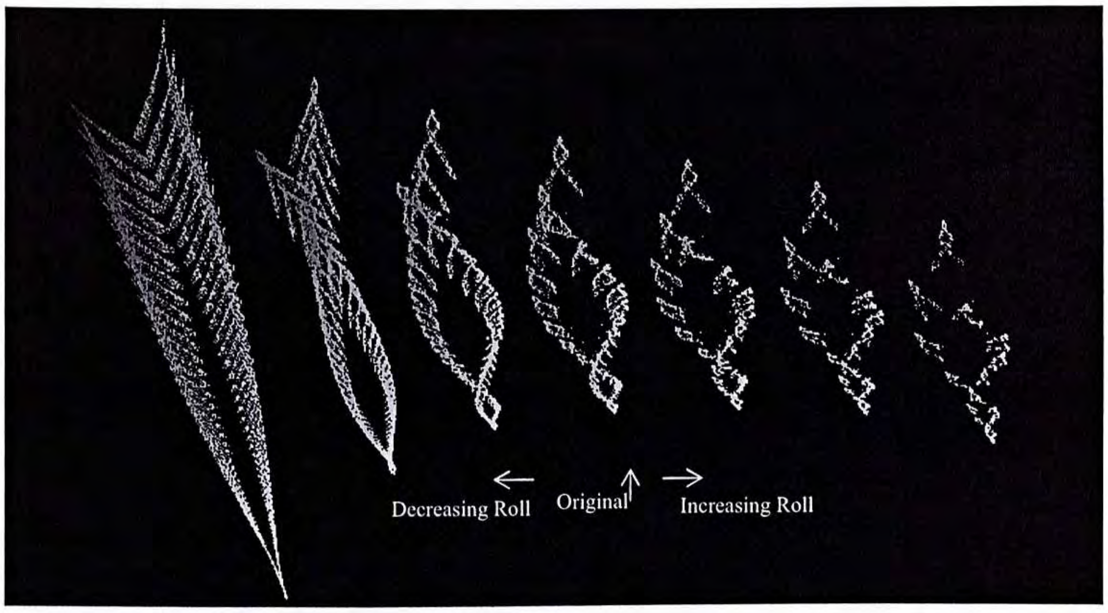


Figure 7.1(d): Effect of changing roll

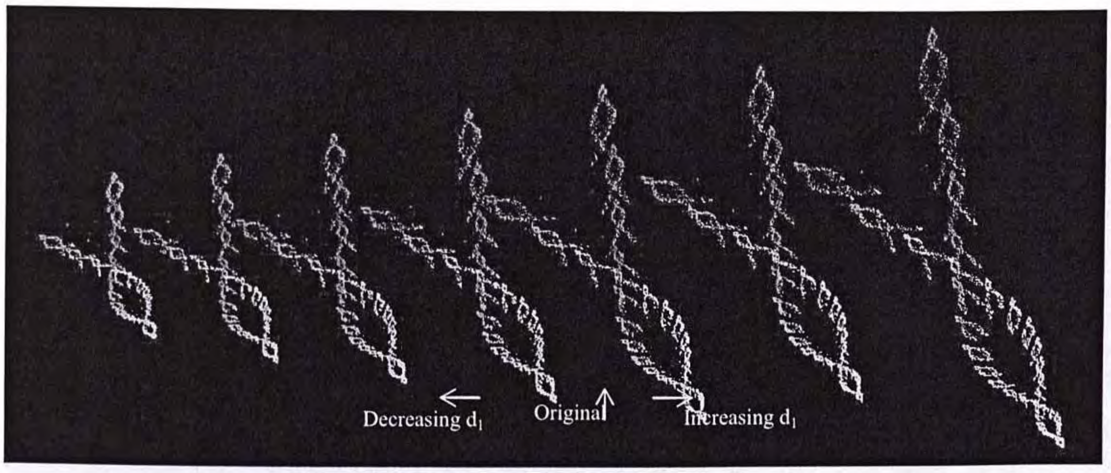


Figure 7.1(e): Effect of changing d_1

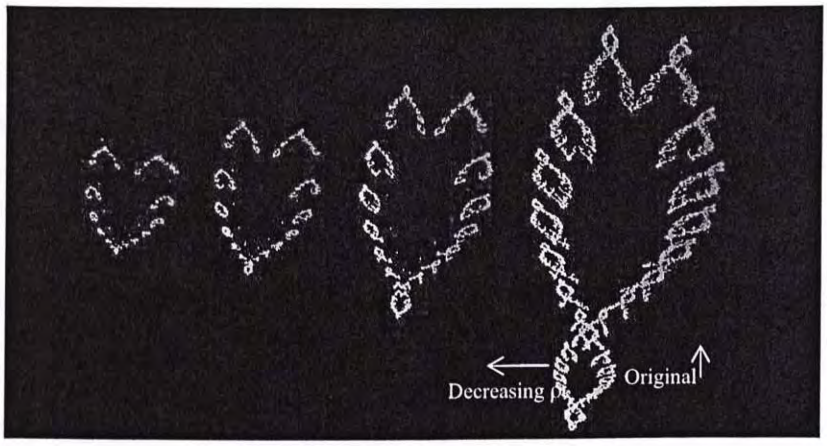


Figure 7.1(f): Effect of changing ρ

7.1.2 Example of user interactive fine-tuning

The example in Figure 7.2 shows a modified fractal pattern 7.1.1-2 and its application as a pendant in jewelry design. The fractal pattern is adjusted based on 7.1.1-1 shown in Figure 7.1(a) by adding $\pi/10$, $2\pi/15$ and $\pi/15$ to pitch, yaw, and roll respectively.

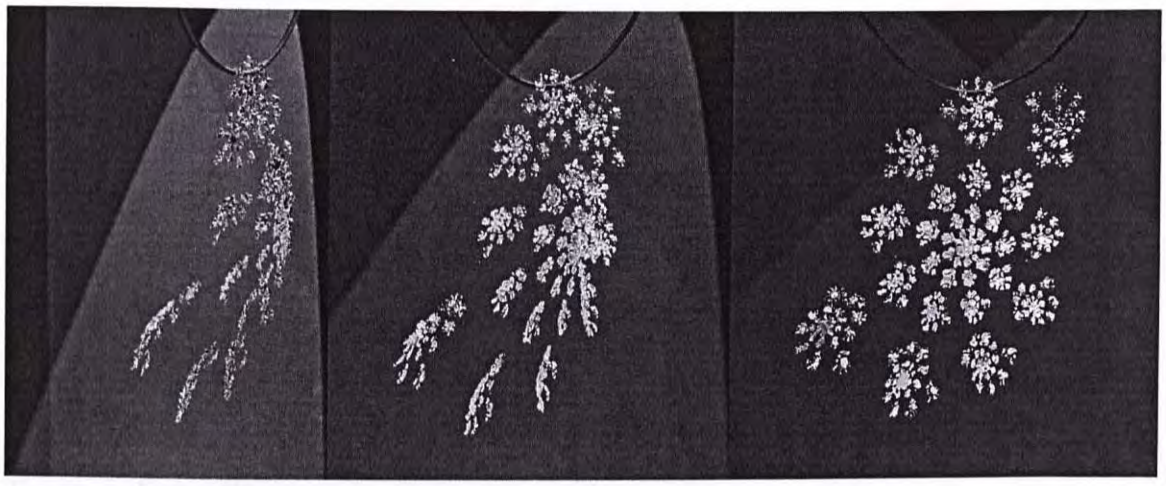


Figure 7.2: Fractal 7.1.1-2, modified based on the fractal 7.1.1-1, rotate clockwisely from left to right

7.2 Visualization and Rendering of 3D Fractals

7.2.1 Volume Point Rendering

The fractals generated through phenotype expression are represented by a set of voxels. Due to the lack of topology and non-uniform distribution characteristic of the fractal representation, the rendering primitives employed for 3D fractals are the volume points, which can be rendered fast and provide users with a coarse image of the fractals.

The rendering process is implemented using OpenGL. A linear gradient coloring scheme is applied for rendering the 3D fractals. The color of the fractal at a specific position is determined by its 3D coordinates. The color of a fractal point is denoted as C(x, y, z) which is the color vector (r, g, b) at the position. The color is decided by C(x, y, z),

$$C(x,y,z)=(r,g,b)=(1,1,1)\times\alpha(x,y,z),$$

$$\text{where } \alpha(x,y,z)=\begin{pmatrix} \frac{x-x_{\min}}{x_{\max}-x_{\min}} & 0 & 0 \\ 0 & \frac{y-y_{\min}}{y_{\max}-y_{\min}} & 0 \\ 0 & 0 & \frac{z-z_{\min}}{z_{\max}-z_{\min}} \end{pmatrix}.$$

The r, g, b channel are associated with the x, y, z coordinate respectively.

A dual-tone gradient color scheme is applied for cases 7.2.1-3, 7.2.1-10~7.2.1-12. The desired color C(x, y, z),

$$C(x,y,z)=(r,g,b)=C_1-C_2\times\alpha(x,y,z)$$

where C_1 and C_2 are the dual base colors.

There are twelve examples shown in Figure 7.3, which are generated in the evolutionary system. The fractals are rendered using linear gradient coloring with 50000 points, and the rendering process takes 5.24 seconds on average. Their corresponding gene codes are listed in Table 7.1.

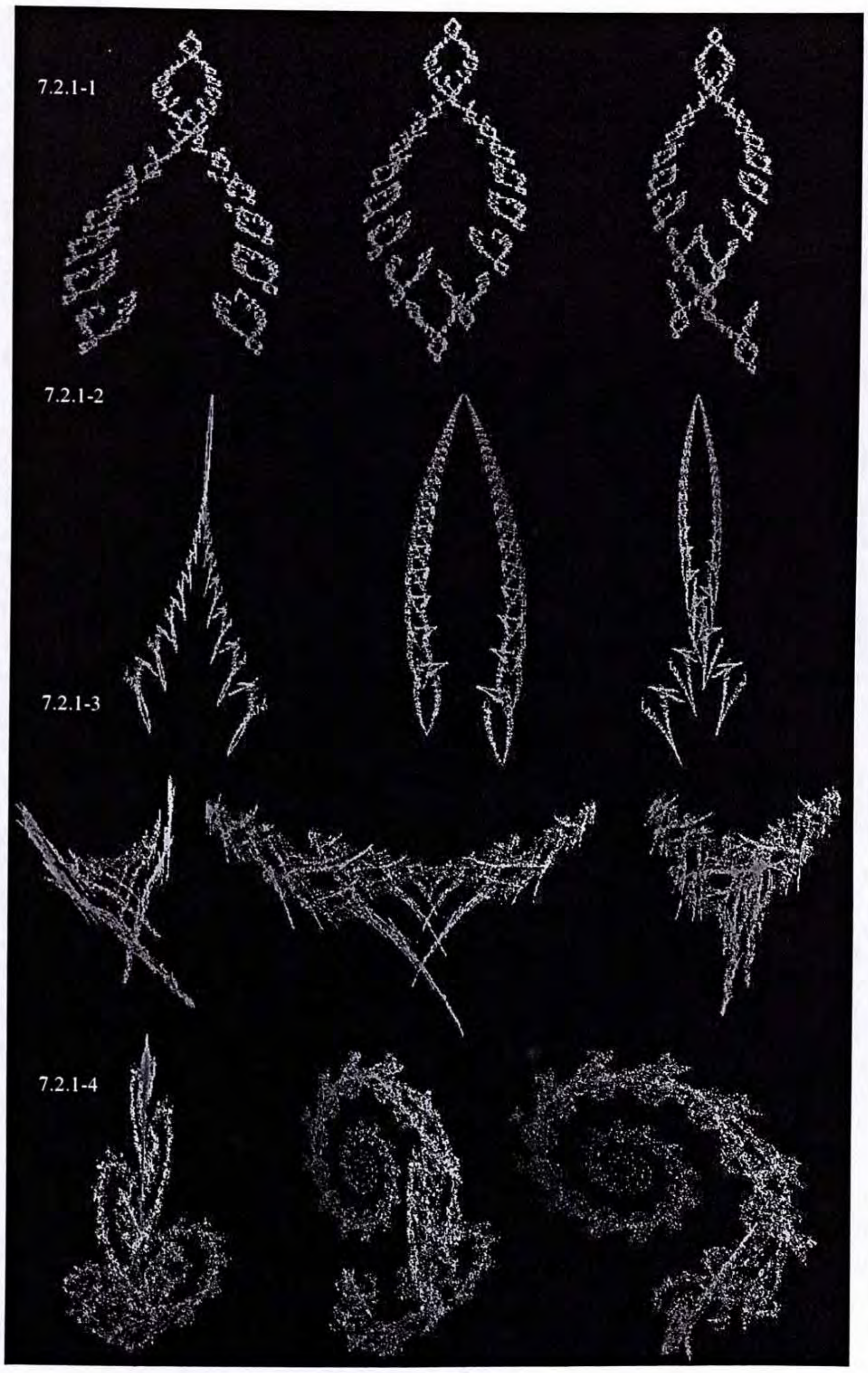


Figure 7.3: Examples of 3D fractals generated in the evolutionary system using linear gradient coloring, rotate counter-clockwisely from left to right

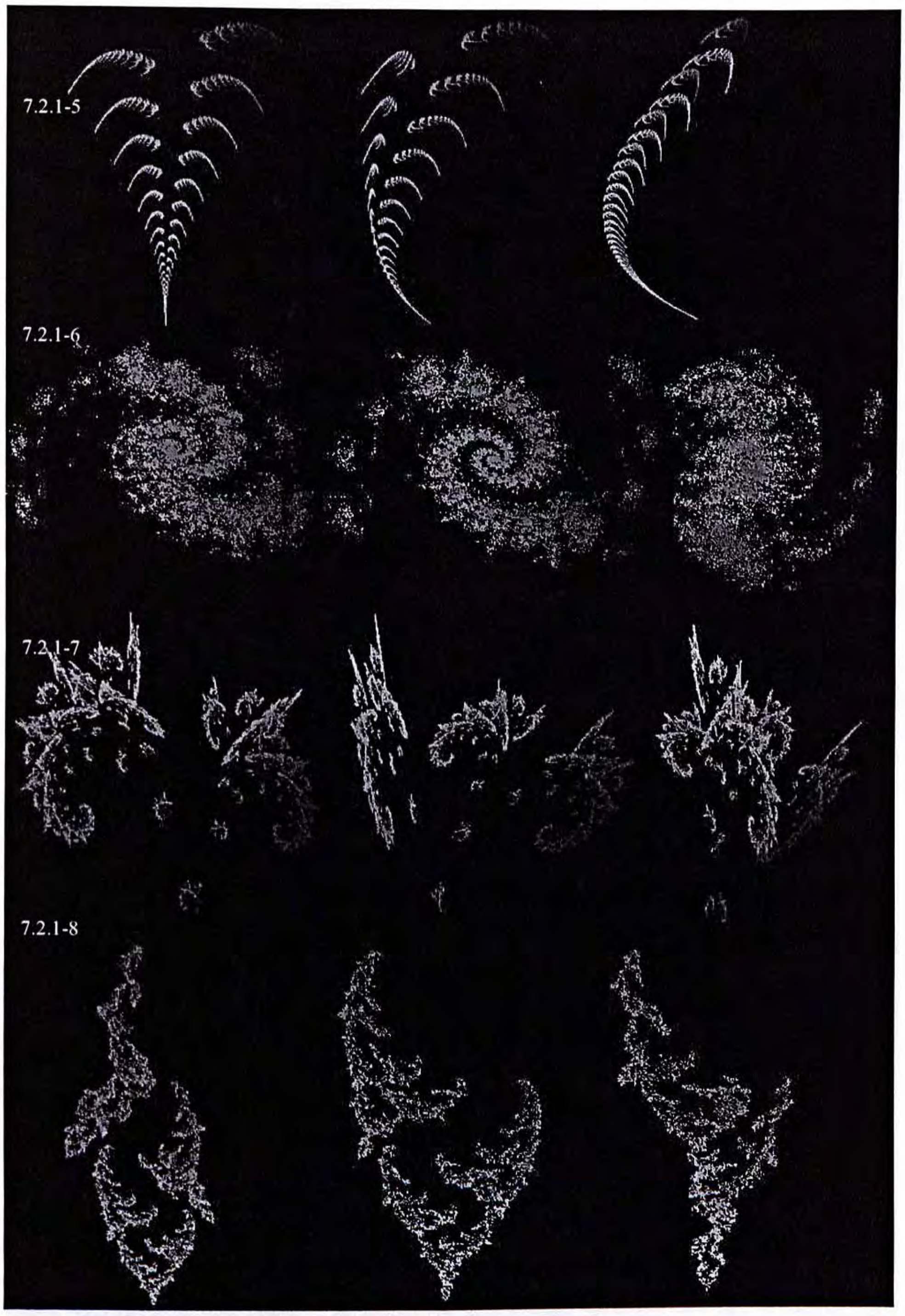


Figure 7.3: (cont'd)

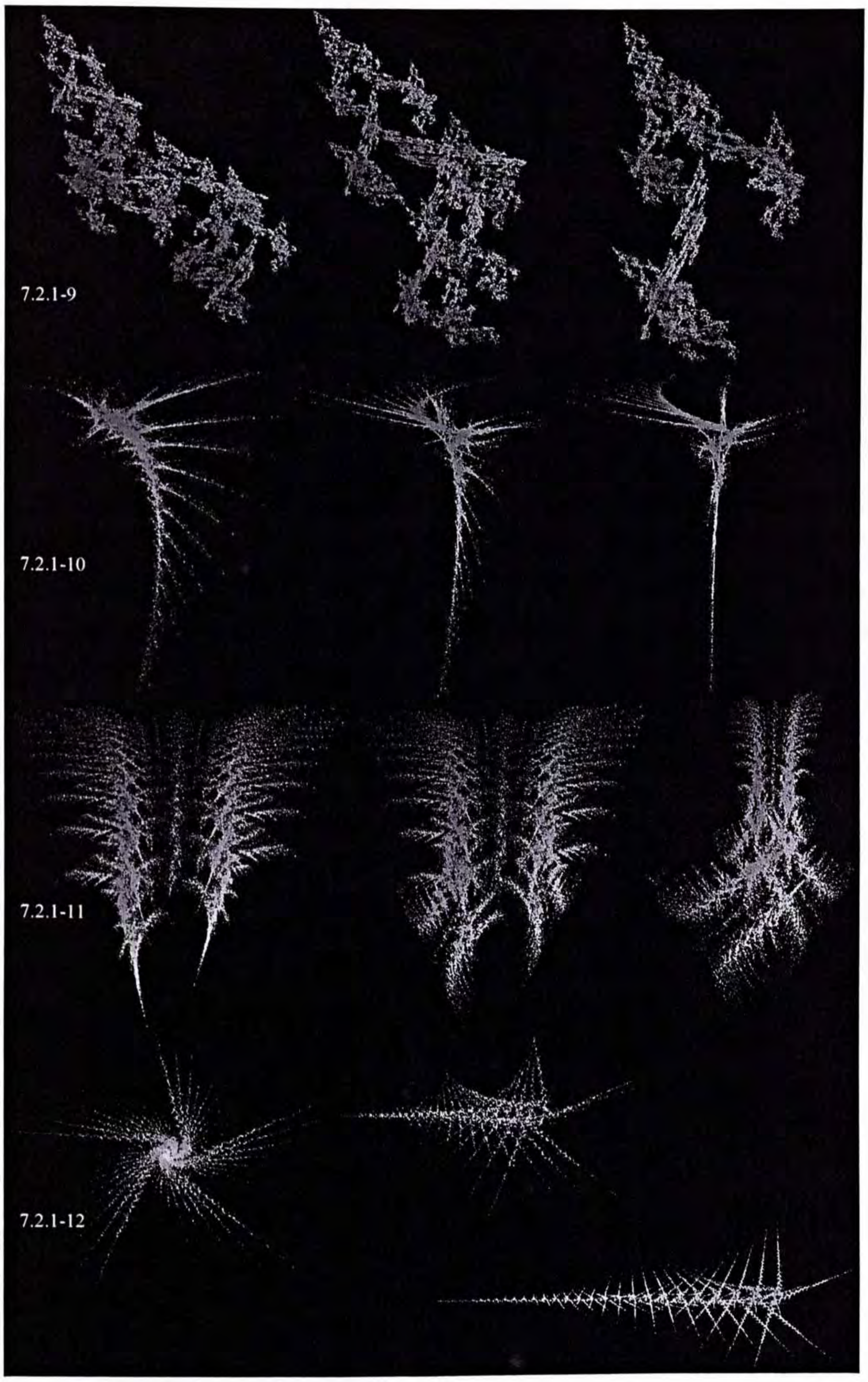


Figure 7.3: (cont'd)

Index	α_p	α_y	α_r	ρ	d_1	d_2	d_3
7.2.1-1	-2.5900	-0.2154	1.9586	0.2793	-0.9179	-1.2826	0.5331
	0.7182	-0.9256	3.0556	0.9677	0.8829	-0.3807	0.0029
7.2.1-2	-2.8438	1.1909	0.7555	0.8832	1.2880	-0.5563	1.3717
	1.7701	1.1045	-0.3307	0.1178	-0.0135	0.7556	0.2456
	1.0339	-2.4817	0.3572	0.2307	-1.3333	-0.9119	0.1188
7.2.1-3	-1.1030	-1.7176	2.5104	0.7542	-1.7473	-0.6002	1.1685
	-0.6791	-3.1013	-0.4611	0.7352	0.9580	0.4858	0.9375
7.2.1-4	2.9765	2.2132	-0.9220	0.5128	2.1764	-0.4438	-0.3157
	1.2971	-0.2559	-1.0080	0.9255	0.4316	0.0300	0.9778
7.2.1-5	-1.6536	-2.1031	2.1542	0.8952	1.1259	0.6706	-0.3158
	2.1971	0.6442	2.0935	0.3530	1.7177	-1.1508	0.1947
7.2.1-6	0.2535	-2.5441	-1.3460	0.2008	1.3614	0.6004	0.8391
	2.9436	-0.4846	2.0856	0.9155	-0.3950	-0.3844	-0.7047
	0.8611	1.2827	-0.9874	0.4224	-0.0935	-1.3934	0.5502
7.2.1-7	1.9352	2.7315	-3.1066	0.1039	2.7786	-0.3765	-0.4451
	2.3017	2.4483	-2.9740	0.6815	0.6308	1.9092	0.9879
	-2.7245	-2.6631	-0.9796	0.6373	-0.6972	0.7471	1.2293
7.2.1-8	0.4098	0.2564	1.2435	0.1343	-0.7845	-0.6577	-0.2600
	-1.4164	1.9393	-1.4556	0.7039	1.0941	-0.0352	0.8753
	1.1650	-2.9849	0.9214	0.7360	-0.4980	0.7629	-0.3071
7.2.1-9	0.7965	0.2119	-1.1741	0.5835	0.0098	-0.9190	0.4921
	-0.0044	0.2654	-1.5714	0.4592	0.6552	-0.3989	2.0563
	0.3056	-0.3511	2.6642	0.7395	-2.7659	1.2394	-1.5298
7.2.1-10	2.2237	1.7187	0.3243	0.7794	0.2761	1.0610	1.7711
	1.2230	-0.1873	0.0150	0.8465	1.3295	1.1357	-0.0139
	-2.1695	0.9688	0.0180	0.2624	1.2817	1.0508	-1.1341
7.2.1-11	2.9523	2.9343	2.9154	0.2095	-0.5115	0.2966	-0.0109
	0.1554	-2.4184	-0.1682	0.5514	0.0268	-0.8664	1.1950
	-0.3223	3.0522	2.3830	0.8780	-0.4476	-1.2096	-0.9740
7.2.1-12	-1.1886	1.2925	2.6916	0.0397	-1.0043	-1.9483	-1.9868
	0.0203	0.1246	1.2773	0.9802	-0.5522	-0.4016	0.9956
	-0.7206	-1.9475	0.4351	0.0965	1.1693	0.0640	0.2261

Table 7.1: The gene code for fractal examples in Figure 7.3

7.2.2 Mesh Surface Rendering

To achieve high quality rendering of fractals, marching cubes is applied to reconstruct the mesh and topology, of the fractals. The reconstruction mesh can then be rendered using standard graphics packages. The Marching cubes technique proposed by Lorensen and Cline [23] extracts a polygonal mesh of an isosurface from a three-dimensional scalar field (also called voxels).

The algorithm works on voxels representing a scalar field. Taking eight neighbor locations at a time (thus forming an imaginary cube), then determines the polygon that represents the part of the isosurface passing through this cube. The individual polygons are then combined to form the desired surface (see Appendix A). The effect of rebuilt fractals is like this:

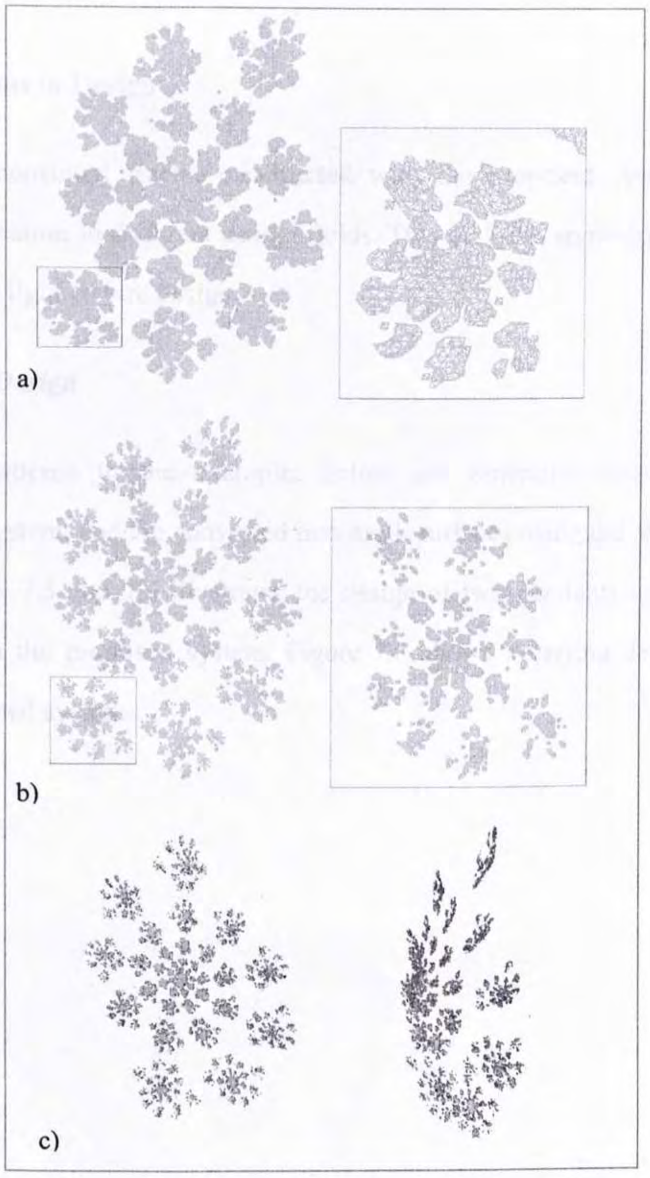


Figure 7.4: Mesh surface rebuilt by Marching Cubes (Based on the fractal 7.1.1-2 shown in Figure 7.2)

a) Wireframe of the reconstructed mesh surface with a grid of 199×165×86. b) Wireframe of the reconstructed mesh surface with a grid of 90×74×39. c) Smooth rendering of the reconstructed mesh.

7.3 Applications in Design

The three dimensional fractal constructed with the proposed system may have potential application in different design fields. This includes applications in jewelry, decoration and light fixture design.

7.3.1 Jewelry Design

The fractal patterns in the examples below are generated with the proposed evolutionary system, and are converted into mesh surfaces using the Marching Cubes method. Figure 7.5 and 7.6 illustrates the design of two pendants using the fractal generated with the proposed system. Figure 7.8 shows a earring design generated with the proposed system.

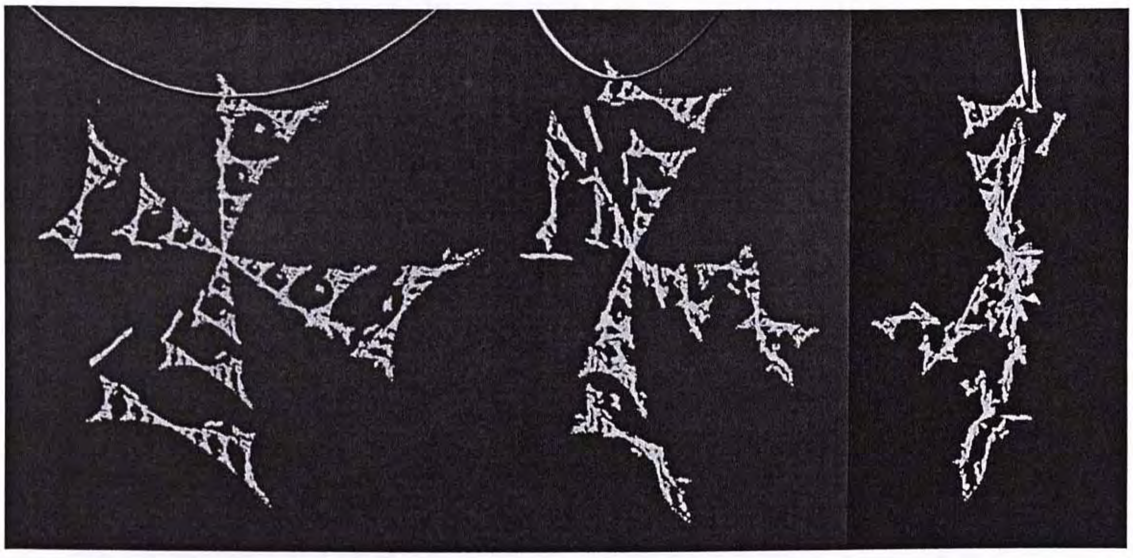


Figure 7.5: Multi-views of 3D fractal jewelry 7.3.1-1, rotate clockwisely from left to right

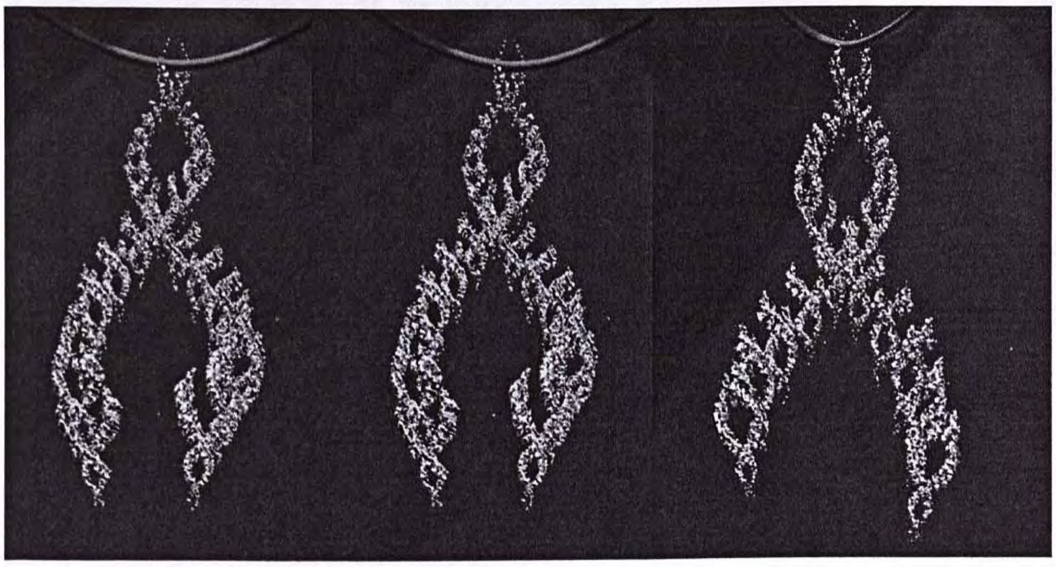


Figure 7.6: Multi-views of 3D fractal jewelry 7.3.1-2, rotate clockwisely from left to right

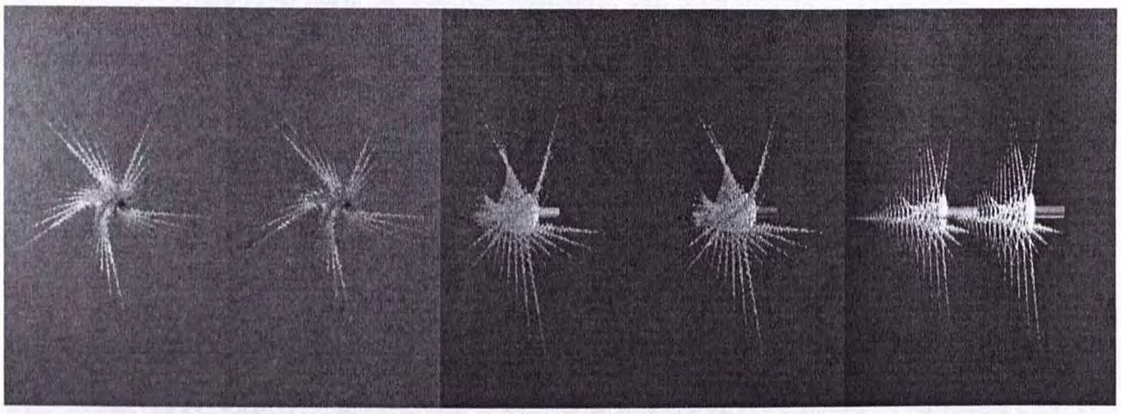


Figure 7.7: Multi-views of 3D fractal jewelry 7.3.1-3, rotate clockwisely from left to right

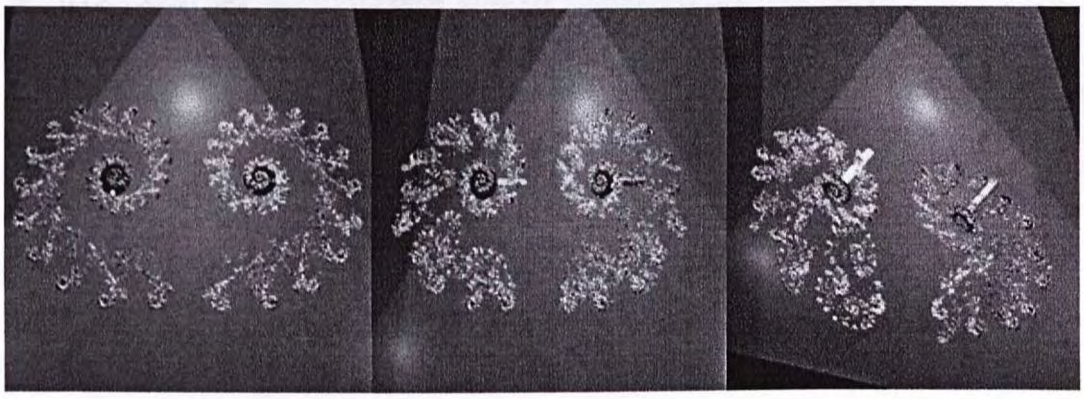


Figure 7.8: Multi-views of 3D fractal jewelry 7.3.1-4, rotate clockwisely from left to right

7.3.2 Decoration Design

Good 3D fractal patterns can be used for all kinds of decorations, such as indoor decorations, ornamental parts of watches, giftware, car, architecture and furniture. A Christmas tree with decorations is shown below. The Christmas tree itself includes two of the fractal with an angle of 90 degree, and the fractal pattern is the first one shown in Figure 7.10. Some parts of the tree are shown respectively in Figure 7.9.

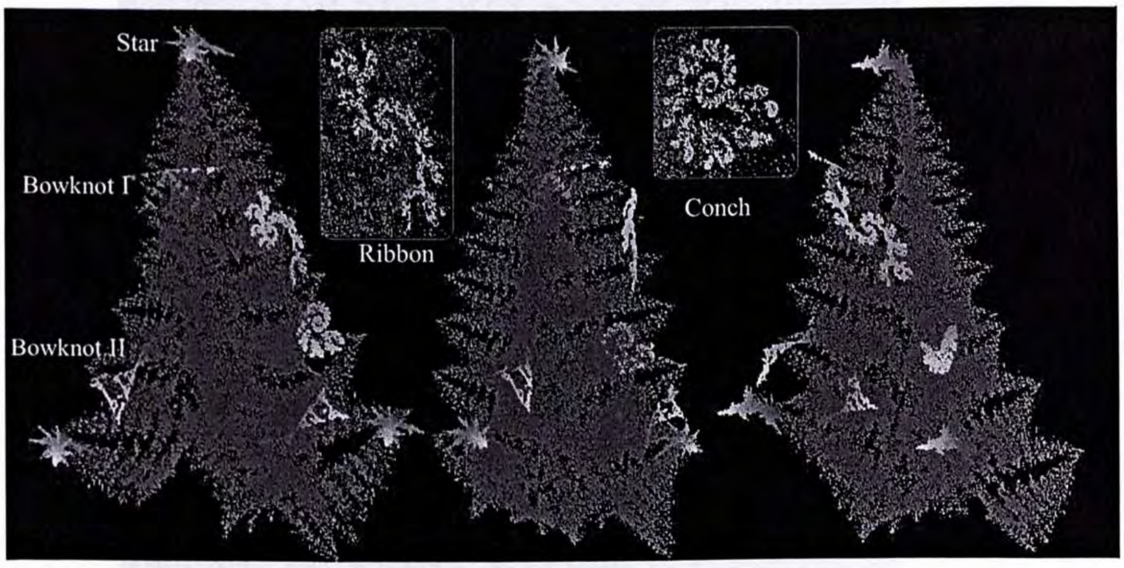


Figure 7.9: A Christmas tree with decoration generated using voxel rendering

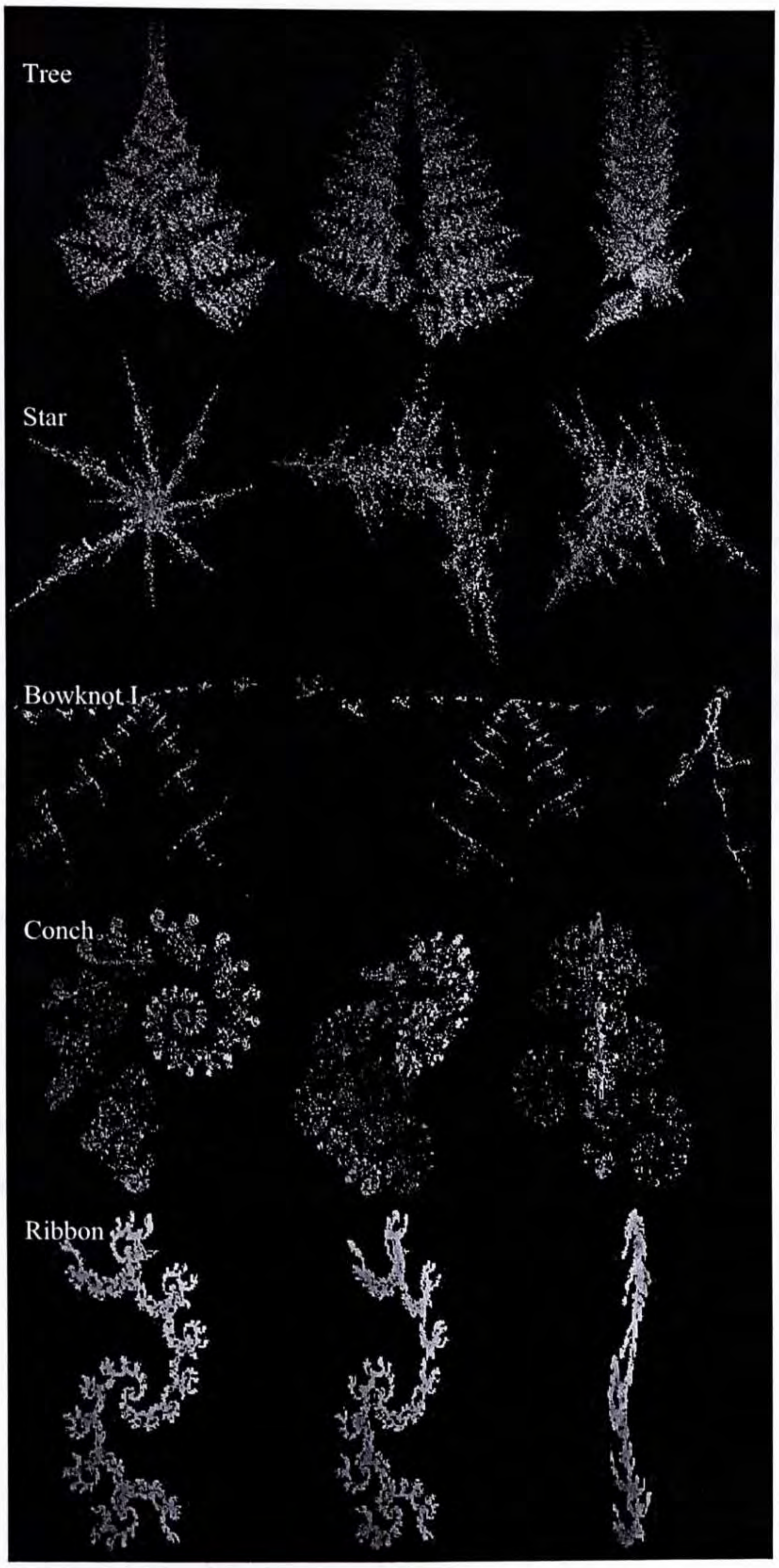


Figure 7.10: Parts of the Christmas tree, rotate counter-clockwisely from left to right

Name	α_p	α_y	α_r	ρ	d_1	d_2	d_3
Tree	-1.1210	2.2344	-0.0970	0.5559	0.5775	-0.1670	0.3658
	-0.5045	3.0315	0.7601	0.9209	0.7531	-0.5818	-0.5489
Conch	1.7688	0.4645	2.9645	0.1149	-1.2266	0.9570	-0.8926
	-0.2650	-0.4257	-1.6297	0.3862	-0.1897	-0.5334	0.2787
	1.5227	1.3276	0.8000	0.9554	-0.3017	-0.9011	-0.7458
Bowknot I	-0.4123	-2.8664	-2.8508	0.2834	-0.4495	0.9077	0.4105
	-1.9922	2.7999	-0.7459	0.1449	-1.5479	2.3696	1.0526
	3.0092	-1.3821	0.4589	0.7591	-0.0958	0.5198	0.4288
Ribbon	1.9939	-1.5441	0.2536	0.5762	-1.8628	-0.6521	-0.2206
	0.3131	-1.0922	-2.8812	0.9214	-0.4542	0.1033	-0.2790
Star	0.2266	-2.1118	-1.6956	0.5096	0.7508	-0.5173	-0.7534
	-3.0754	2.2943	-3.0442	0.8928	0.5002	-0.5592	0.9258
Bowknot II	-2.8638	-1.8174	-0.9384	0.4143	1.0163	0.2510	0.5775
	0.8404	-1.8456	-2.2571	0.1471	-2.6759	0.7920	0.2907
	-3.1260	-1.3764	0.4473	0.6898	-0.0052	0.5922	0.4090

Table 7.2: The gene code for fractal examples in Figure 7.9

7.3.3 Light Fixture Design

By considering the voxels representing a 3D fractal as locations of light emitting diode (LED), patterns generated with the proposed technique can be used for light fixture design. A virtual scene of a light fixture with decoration effect in a room is shown in Figure 7.11.

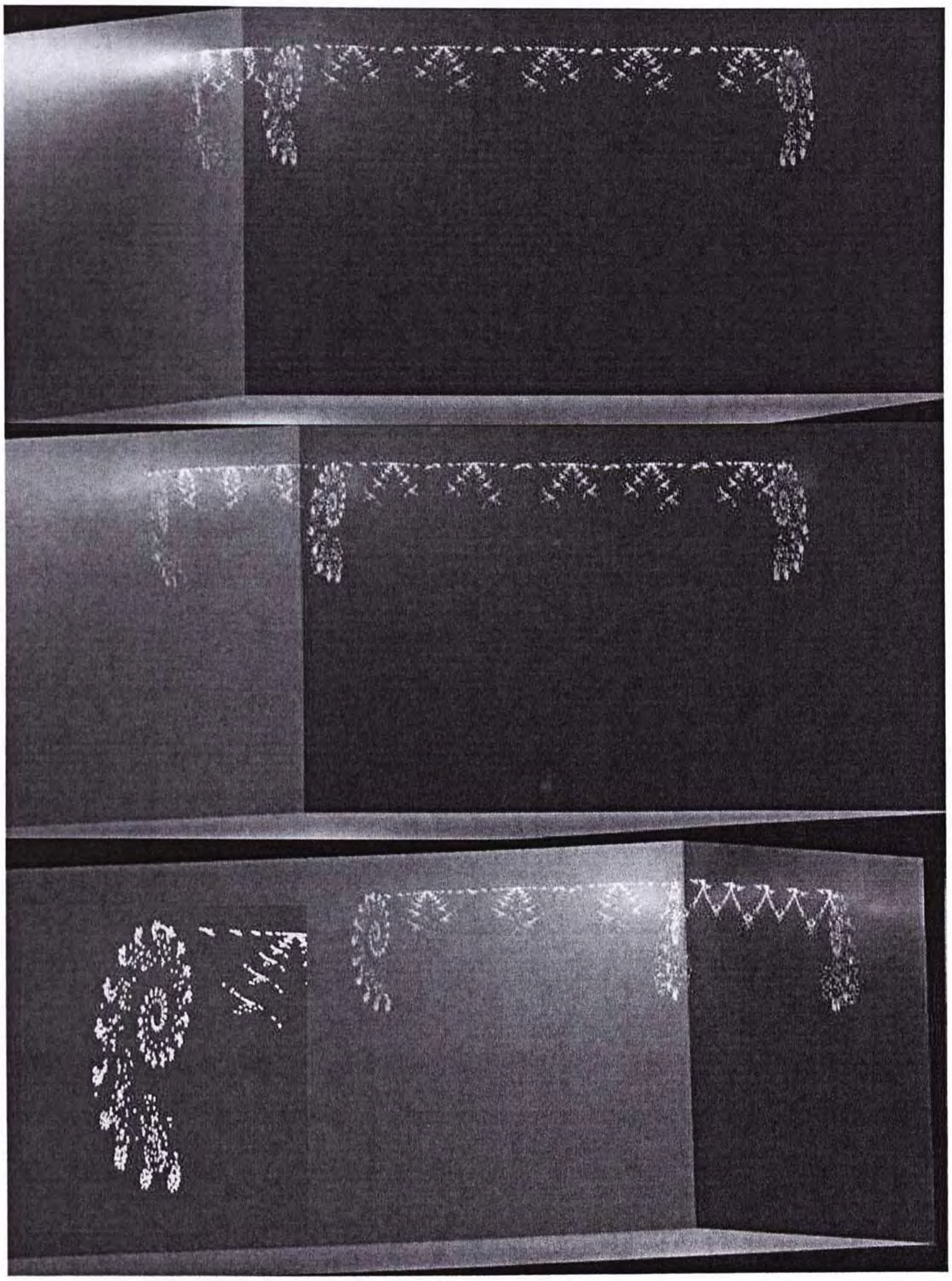


Figure 7.11: Example of light fixture design. The scene is rotated clockwisely from top view to bottom view and the bottom-left corner in third frame is the enlarged view of the leftmost light fixture

8. CONCLUSIONS and FUTURE WORK

8.1 Conclusions

In this research, an approach for automatic construction of 3D fractals is proposed. A new three-dimension IFS formula, called Fractal Transformation IFS (FT IFS), with revised random iteration algorithm is put forward which can improve the efficiency in the generation of fractals. The parameters in the formula can be easily controlled by designers and thus allow interaction in the generation process.

A revised evolutionary algorithm is developed to further improve the system efficiency. Crossover, arithmetic Gaussian mutation and inferior elimination are applied during the evolutionary process. Fitness function is built up using linear regression based on 1641 cases. Parameters, borrowed from dynamics and chaotic system, such as Capacity dimension, Correlation dimension and Largest Lyapunov Exponent, constitute the 3D fractal fitness function.

Finally the fractal transform method for controlling fractal patterns is tested with different design applications. The visualization and rendering of three dimensional fractals are discussed. Potential applications are explored and examples are included.

The proposed FT IFS method has some limitations. The running time of generating and rendering a fractal pattern is 12.39 seconds on average, and this restricts the instant user interaction. Due to the randomness of the constructing process, it is possible that the generated fractals look like a mass over the space.

8.2 Future Work

IFS does not limit the dimension of the system. The dimension of the proposed FT IFS is three, and it is possible to extend it to more dimensions, 4-dimension or N-dimension. Some researchers [43] adopted three dimension IFS for the 2d fractal art, and stored the color information in the third dimension. Due to the transforming property of the proposed FT IFS, fractal shape transform smoothly as the parameters change. It could be extended to four dimensional space to produce animation, and the fourth dimension can store the time function.

More aesthetic rules of 3d fractals can be developed to classify the aesthetics value more accurately and further enhance the percentage of the generated visually appealing fractal. Factors, like symmetry, golden ratio, complexity, can be taken into consideration, and how these factors are quantified for 3d fractals could be investigated.

Stylistic aesthetic evaluation equation and evolutionary system for specific designer could be developed. Special fitness function can be formulated by collecting data when the designers are selecting the fractal pattern during the process. Other criteria that should be taken into account in design evaluation are color and manufacturability.

Work may also be done to explore various application fields of the proposed approach and identify any scope for improvement.

BIBLIOGRAPHY

[1] Aks, D.; Sprott, J.C., Quantifying aesthetic preference for chaotic patterns, Empirical Studies of the Arts, 1996, 14, 1–16.

[2] Barnsley, M.; Demko, S.: Iterated Function Systems and the Global Construction of Fractals, Proc. Royal Soc. of London, 1985, A 399(1817), 243-275.

[3] Barnsley, M.: Fractals Everywhere, 2nd ed., Academic Press, San Diego, USA, 1993.

[4] Bentley, P.J.; Wakefield, J.: The evolution of solid object designs using genetic algorithms, Modern Heuristic Search Methods, 1996, pp. 199–215.

[5] Bentley P.J.: Aspects of evolutionary design by computers, Proceedings of Advances in Soft Computing--Engineering Design and Manufacturing, London, 1999, 99～118.

[6] Bentley, P.J.: Evolutionary design by computers, Morgan Kaufmann Publishers, UK, 1999, ISBN: 1-55860-605-X.

[7] Berkowitz, J.: Fractal Cosmos: The Art of Mathematical Design, Amber Lotus, Portland, 1998.

[8] Berlyne, D.E.: Ends and means of experimental aesthetics, Canadian Journal of Psychology, 1972, Vol 26(4), 303-325.

[9] Cho, S.-B.: Towards creative evolutionary systems in interactive genetic algorithm, Applied Intelligence, 2002, 16(3), 129 – 138.

[10] Eckert, C., Kelly, I.; Stacey, M.: Interactive generative systems for conceptual design: An empirical perspective, Artificial Intelligence for Engineering Design,

Analysis and Manufacturing, 1999, 13(4), 303–320.

[11] Eckmann, J.P.; Ruelle, D.: Ergodic Theory of Chaos and Strange Attractors, Rev. Mod. Phys., 1985, 57(3), 617-656.

[12] Falconer, K.J.: Fractal Geometry: Mathematical Foundations and Applications, John Wiley & Sons Ltd, New York, USA, 2003, ISBN 0-470-84862-6.

[13] Fechner, G. T.: Vorschule der Äesthetik, Breitkopf & Härtel, Leipzig, 1876.

[14] Galanter, P.; Levy, E.K.: Complexity, the Leonardo gallery, Leonardo, 2003, 36(4), 259 – 267.

[15] Gero, J.S.: Creativity, emergence and evolution in design, Knowledge-Based Systems, 1996, 9(7), 435 – 448.

[16] Grassberger, P.; Procaccia, I.: "Measuring the Strangeness of Strange Attractors", Physica D: Nonlinear Phenomena, 1983, 9 (1-2), 189-208.

[17] He, X.G.; Lau, K.S.: On a generalized dimension of self-affine fractals, Math. Nachr, 2008, 281(8), 1142-1158.

[18] Jacob, C.: A Power Primer, Psychological Bulletin, 1992, 112(1), 55-159.

[19] Fractal Flame, http://en.wikipedia.org/wiki/Fractal_flame, 2009.

[20] Kerry Mitchell, The Fractal Art Manifesto, https://www.fractalus.com/info/manifesto.htm, 2009.

[21] Koile, K.: Formalizing abstract characteristics of style, Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 2006, 20(3), 267–285.

[22] Kruger, A.: Implementation of a fast box-counting algorithm, Computer Physics Communications, 1996, 98(1-2), 224-234.

[23] Lorensen, W.E.; Cline, H.E.: Marching Cubes: A high resolution 3D surface construction algorithm, Computer Graphics, 1987, 21(3), 163-169.

[24] Mandelbrot, B.B.: The Fractal Geometry of Nature, W.H. Freeman and Company, 1982, ISBN 0-7167-1186-9.

[25] Merkwirth, C.; Parlitz, U.; Wedekind, I.; Lauterborn, W.: TSTOOL MatLAB toolbox, http://www.physik3.gwdg.de/tstool/, 2002.

[26] Mitina, O.V.; Abraham, F.D.: The use of fractals for the study of the psychology of perception: psychophysics and personality factors, a brief report, International Journal of Modern Physics C, 2003, 4(8), 1047–1060.

[27] Moles, A.A.: Information Theory and Esthetic Perception, University of Illinois Press, Urbana, 1968.

[28] Ning, L: Fractal imaging, Academic Press, 1997, ISBN 0-1245-8010-6.

[29] North, A.C.; Hargreaves, D.J.: Experimental aesthetics and everyday music listening, The social psychology of music, 1997, pp. 84-103.

[30] Articles for deletion/Sterling Fractal, http://en.wikipedia.org/wiki/Wikipedia_talk:Articles_for_deletion/Sterling_F ractal, 2009.

[31] Poirson, E.; De'pince', P.; Petiot, J.-F.: User-centered design by genetic algorithms: application to brass musical instrument optimization, Engineering Applications of Artificial Intelligence, 2006, 20(4), 511–518.

[32] Remko, S., Rens, B.: Computational esthetics, Informatie en Informatiebeleid 11(1), 54–63, 1993.

[33] Rosenman, M.A.: The generation of form using an evolutionary approach,

Evolutionary Algorithms in Engineering Applications, 1996, pp. 69–85.

[34] Rowland, D.; Biocca, F.: Evolutionary co-operative design between human and computer: implementation of "the genetic sculpture park.", Proc. 5th Symp. Virtual Reality Modeling Language (Web3d-Vrml), VRML '00, 2000, pp. 75–79.

[35] Scherer K.: Example of Fractal Art, http://en.wikipedia.org/wiki/File:Www_y23_com--fractal----Lg_Z010121Z.jpg, 2009.

[36] Scott D.: The electric sheep, ACM SIGEVOlution, 2006, 1(2), 10-16.

[37] Scott D.: The Fractal Flame Algorithm, http://flam3.com/flame.pdf, 2004.

[38] Scott, D.; Ralph, A.; Pablo, V.; Frederick, D. A.; Julian, C. S.: The Aesthetics and Fractal Dimension of Electric Sheep, International Jornal of Bifurcation and Chaos, 2008, 18(4), 1243 - 1248.

[39] Sims, K.: Artificial evolution for computer graphics, Computer Graphics, 1991, 25(4), 319-328.

[40] Spehar, B.; Clifford, C.W.G.; Newell, B.R.; Taylor, R.P.: Universal aesthetic of fractals, Computer & Graphics, 2003, 27(5), 813–820.

[41] Fractal, http://en.wikipedia.org/wiki/Fractal, 2009.

[42] Sprott, J. C.: Strange Attractors: Creating Patterns in Chaos, M&T, NY, USA, 1993.

[43] Sprott, J.C.: Automatic Generation of Iterated Function Systems, Computer & Graphics, 1994, 18(3), 417-425.

[44] Crossover in Genetic Algorithm,

http://en.wikipedia.org/wiki/Crossover_(genetic_algorithm), 2009.

[45] Staudek, T.: Computer-aided aesthetic evaluation of visual patterns, ISAMA-BRIDGES Conf. Proc., 2003, pp. 143–149.

[46] Todd, S.; Latham, W.: Evolutionary Art and Computers, Academic Press, London, 1992.

[47] Todd, S.; Latham, W.: The mutation and growth of art by computers, Evolutionary Design by Computers, 1999, pp. 221–250.

[48] Unemi, T.: Simulated breeding—a framework of breeding artifacts on the computer, The International Journal of Systems & Cybernetics, 2003, 32(1/2), 203 – 220.

[49] Vince, J.: Mathematics for Computer Grahics, 2nd ed., Springer-Verlag, 2006.

[50] Wannarumon, S.; Bohez, E.L.J.: A new aesthetic evolutionary approach for jewelry design, Computer-Aided Design & Applications, 2006, 3(4), 385–394.

[51] Wannarumon, S.; Bohez E.L.J.; Annanon, K.: Aesthetic evolutionary algorithm for fractal-based user-centered jewelry design, AI EDAM, 2008, 22(1), 19-39.

[52] The Marching Cubes, http://users.polytech.unice.fr/~lingrand/MarchingCubes, 2009.

[53] Dewdney, A.K.: A computer microscope zooms in for a close look at the most complicated object in mathematics, Scientific American, 1985, Aug, 16–24.

[54] Peitgen, H. O.; Richter, P. H.: The Beauty of Fractals: Images of Complex Dynamical Systems, Springer, Berlin, 1996.

[55] Barnsley's Fern, http://en.wikipedia.org/wiki/Iterated_function_system, 2009.

[56] S.C. Soo; K.M. Yu: Tool-Path Generation for Fractal Curve Making, The International Journal of Advanced Manufacturing, 2002, 19(1), 32-48.

[57] S.C. Soo; K.M. Yu, W.K. Chiu: Modeling and fabrication of artistic products based on IFS fractal representation, Computer-Aided Design, 2006, 38(7), 755-769.

[58] W.J. Pang; K.C. Hui: Interactive Evolutionary 3D Fractal Modeling with Modified IFS, Computer-Aided Design and Applications, 2009, 6(1), 55-67.

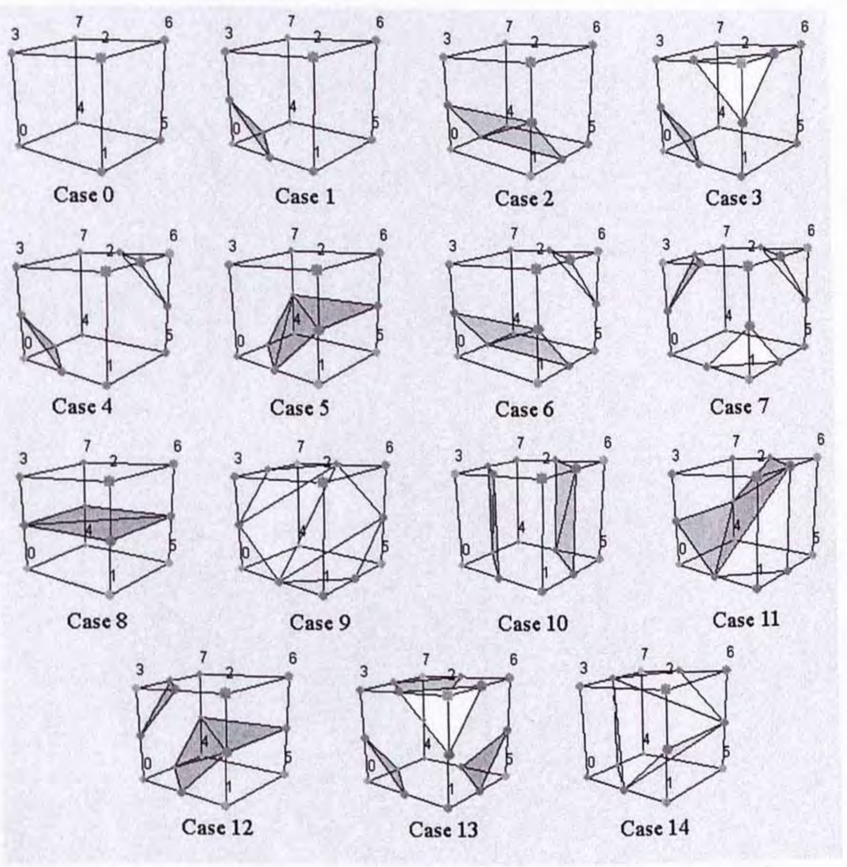
Appendix

Marching Cubes Method

Marching Cubes is an algorithm for rendering isosurfaces in volumetric data. The basic notion is that we can define a voxel (cube) by the pixel values at the eight corners of the cube. If one or more pixels of a cube have values less than the user-specified isovalue, and one or more have values greater than this value, we know the voxel must contribute some component of the isosurface. By determining which edges of the cube are intersected by the isosurface, we can create triangular patches which divide the cube between regions within the isosurface and regions outside. By connecting the patches from all cubes on the isosurface boundary, we get a surface representation. This algorithm is often used to extract the surface of medical organs. It provides a fast and easy way to get from serial sections to a complete 3D object [52].

The main concept of Marching Cubes is to create an index to a precalculated array of 256 possible polygon configurations (28 = 256) within the cube, by treating each of the 8 scalar values as a bit in an 8-bit integer. All these cases can be generalized in 15 families by rotations and symmetries, see figure below. If the scalar's value is higher than the isovalue (i.e., it is inside the surface) then the appropriate bit is set to one, while if it is lower (outside), it is set to zero. Finally each vertex of the generated polygons is placed on the appropriate position along the cube's edge by linearly interpolating the two scalar values that are connected by that edge.

The gradient of the scalar field at each grid point is also the normal vector of a hypothetical isosurface passing from that point. Therefore, we may interpolate these normals along the edges of each cube to find the normals of the generated vertices which are essential for shading the resulting mesh with some illumination model.



The algorithm of Marching Cubes is as follow:

1, Read four slices into memory;

2, Create a cube from four neighbors on one slice and four neighbors on the next slice;

3, Calculate an index for the cube;

4, Look up the list of edges from a pre-created table;

5, Find the surface intersection via linear interpolation;

6, Calculate a unit normal at each cube vertex and interpolate a normal to each triangle vertex;

7, Output the triangle vertices and vertex normals.