Hypercomplex Iterations, Distance Estimation and Higher Dimensional Fractals

Yumei Dang, Louis H. Kauffman and Dan Sandin

To Yumei's father, Ge Dang, and to the memory of Lou's mother, Alice Fisher Kauffman

Contents

Acknowledgements

We particularly thank Joe Insley and John Hart for many discussions and common work together. We thank Aushra Abouzeid and Beth Cerny for infinite amounts of work in preparing the manuscript and designing the accompanying CD-ROM. We thank Floyd Hanson for help in obtaining support that made this project possible, and we thank the NCSA and the PSC for the use of their computer facilities. The second author thanks the National Science Foundation for support under grant DMS-2528707.

Preface

This book is an exploration of methods to visualize fractals in spaces of dimension two, three, and higher. Our basic method involves estimating the distance to a fractal that is generated through the iteration of a complex or hypercomplex function of many variables. The book is based on Yumei Dang's doctoral dissertation and on previous work in fractal geometry, including the work of John Hart, Louis Kauffman, and Dan Sandin.

Julia sets of quadratic functions, as well as many other deterministic fractals, exist in spaces of higher dimension than the complex plane. Visualization of hypercomplex Julia sets is much more difficult than visualization in the complex plane, and the efficiency of the algorithms becomes a significant issue. Several algorithms have been used to view these structures that either required a large amount of storage space or used a low resolution rendering method. One ray-tracing algorithm using distance estimation was given in which the estimation for quaternion space works well, but the algorithm was not justified mathematically. During our study and research, a mathematical justification of this algorithm in both the original complex case and its higher dimensional analogues was discovered.

The distance estimation formula provides an approximation of the distance from a point outside of a Julia set to the Julia set itself. Since Julia sets are fractals, it is extremely difficult to get an exact evaluation of the distance from a point to a Julia set. Therefore, the lower bounds and upper bounds become very important for the purpose of visualization. The distance estimate is a delicate matter to establish even in the case of Julia sets over the complex numbers. It depends on the Riemann Mapping Theorem and a number of special results in complex analysis. It has been verified empirically that in most cases the exact analogue of this distance estimate works very well for quaternionic Julia and Mandelbrot sets in three and four dimensions. The justification of this experimental work demands both new theoretical work and a more fine-grained approach to the experimentation.

On the theoretical side, we pursued a program to analyze the distance estimate using theorems in analysis over the quaternions. Then, we found a mathematical justification and description for the distance estimate in higher dimensional spaces by using a geometric method. This new method provides a fundamental theoretical background for the visualization of deterministic fractals.

To complement this theoretical base, we pursue optimization techniques to improve the distance estimation algorithm and corresponding rendering methods on powerful computer systems. Since images of hypercomplex fractals are time consuming to create, the flexibility of the experiments is limited. This is the primary mathematical reason for wanting the best supercomputers to help in this problem. In more than two years of experiments on the Alpha-Cluster at PSC, the SGI Power Challenge at NCSA and the virtual reality environment CAVE at EVL, finer and more flexible results have been obtained.

We expect that a high-quality and flexible source of images of higher dimensional Julia sets, Mandelbrot sets and other deterministic fractals will raise many new mathematical questions.

The book is organized in four parts. In Part I, Chapter 1 gives an introduction to hypercomplex iterations and Chapter 2 discusses deterministic fractals and distance estimation.

Part II, covered by Chapters 3 through 5, provides mathematical background on the classical method for estimating the distance to complex fractals. Chapter 4 provides the reader with a wealth of interesting lore about quaternionic analysis, though its eventual conclusion is pessimistic about the application of these techniques to hypercomplex distance estimation. Chapter 5 is an interlude about the relationships among quaternions, topology, and quantum mechanics.

Part III begins the serious analysis of hypercomplex distance estimation. In Chapter 6, we show how to estimate the distance to quaternion Mandelbrot sets by reducing the problem to the complex case. In Chapter 7 we give our new methods for distance estimation, outlined in Chapter 1, and apply them to specific cases of higher dimensional iterations.

Part IV discusses in detail issues involved in generating, modelling and rendering hypercomplex fractals. Chapters 8 and 9 present two different algorithms for modelling deterministic fractals: inverse iteration and ray-tracing by distance estimation. A brief description of other visualization methods (including animation) on platforms such as the CAVE and ImmersaDesk, supercomputing techniques, and new data structures are described in Chapter 10.

This book was written using the LaTeXdocument processing system. The 2D illustrations were created with Silicon Graphics IRIS-4D workstations. The 3D illustrations were created on SGI Power-Challenge and IRIS-4D workstations.

Part I

Introduction

Chapter 1

Hypercomplex Iterations in a Nutshell

This book is an exposition of research related to the rendering of images of fractals in dimensions three and higher. We concentrate on fractals that are generated by iterating functions analogous to those used to make the well-known Mandelbrot and Julia sets in the complex plane. Thus we study quaternionic and hypercomplex iterations (defined below) and we use a technique of distance estimation that generalizes a method well-known in the case of the complex plane.

Our key result is a precise formulation and justification of that distance estimation algorithm. Our justification of the algorithm solves a question that has been open for some time, since it was empirical knowledge among some of us that the method did produce striking visual results, and it was conjectured that it was mathematically correct. In this introduction we will outline our approach to the distance estimation. In the body of the book we give the analytic background for the classical approach to this method and a concise introduction to quaternionic analysis in the hope that this work will be of use for other problems in this field. As the reader will see, our eventually successful approach to distance estimation is quite elementary and easily understood with a minimum of background.

After a general introduction, we recommend that the reader browse through the book and find those sections that are most relevant to his or her interest. In particular, it is quite possible to go directly to Part III of the book dealing with Hypercomplex Iterations, skipping the intricate background of complex and quaternionic analysis.

The rest of this introduction will be a quick tour of the themes and ideas of the book. To begin, let us recall the form of the quaternionic number system. Just as the complex numbers $a + bi$ (with a and b real) are generated by the real unit 1 and the imaginary unit i, the quaternions $t + ai + bj + ck$ (with t, a, b, c real) are generated by the real unit 1 and the imaginary units i, j, k. These units satisfy the following equations

$$ij = -ji = k,$$

$$jk = -kj = i,$$

$$ki = -ik = j,$$

$$i^2 = j^2 = k^2 = -1.$$

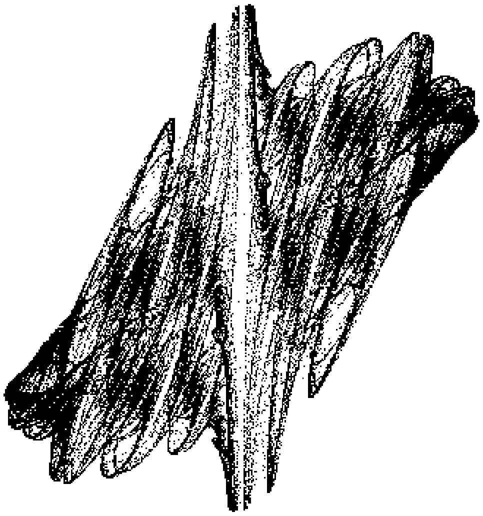

Figure 1.1: $A = -0.25, B = 0.75, \theta = 0$

Note that it follows that $ijk = -1$. William Rowan Hamilton, who discovered the quaternions, was so taken by them that he inscribed the equations

$$i^2 = j^2 = k^2 = ijk = -1$$

on the side of a bridge on the day of the discovery. The multiplication of quaternions is associative, a kind of miracle that cannot be repeated for analogous systems such as the vector cross product, or generalizations such as the Cayley numbers (see Section 7.2).

The quaternions have many remarkable properties. First there are formal properties of the multiplication. If $w = t + ai + bj + ck$ with a, b, c, t real numbers, then we define the *conjugate* of w to be the quaternion $\overline{w} = t - ai - bj - ck$. It is easy to check that

$$w\overline{w} = t^2 + a^2 + b^2 + c^2.$$

Hence, non-zero quaternions have multiplicative inverses. In this sense, the quaternions generalize all the properties of complex numbers except for commutativity.

We call a quaternion with $t = 0$ a *pure* or *purely imaginary* quaternion. Letting

$$u = ai + bj + ck,$$

it is easy to compute that

$$u^2 = -(a^2 + b^2 + c^2).$$

Thus if u is a pure quaternion with length one (i.e., $a^2 + b^2 + c^2 = 1$) then $u^2 = -1$. Among the pure quaternions, there is an entire two-dimensional sphere's worth of square roots of negative unity. There is one square root of minus one for each direction in the three dimensional space of pure quaternions.

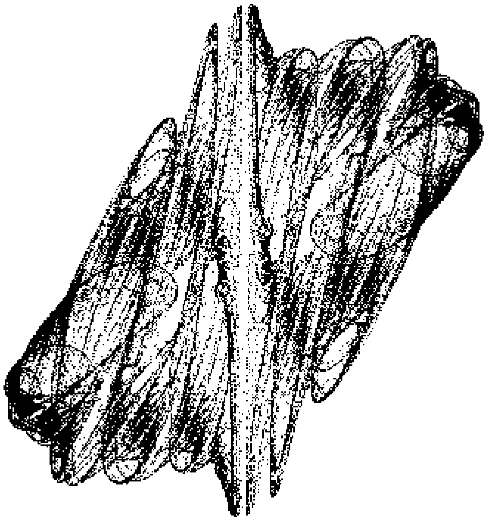

Figure 1.2: $A = -0.25, B = 0.75, \theta = 45$

It is worth noting that if u and v are two pure quaternions then

$$uv = -u \cdot v + u \times v$$

where $u \cdot v$ denotes the dot product of vectors in three-space and $u \times v$ denotes the cross product of vectors in three-space. In this notation, points in three-space are pure quaternions and the addition of a scalar connotes a coordinate in the fourth dimension.

It is convenient to write a quaternion in the form $A + Bu$ where A and B are real numbers and u is a unit length pure quaternion. Then $u^2 = -1$ and powers of $A + Bu$ take the same form as powers of complex numbers (since u taken alone is indistinguishable from any other square root of minus one). Thus if $z = A + Bu = R\cos(\theta) + R\sin(\theta)u$, then

$$z^n = R^n \cos(n\theta) + R^n \sin(n\theta)u$$

by DeMoivre's formula for powers of complex numbers. This means that we can define power mappings and also n-th roots in the quaternions.

This method allows us to define a partial multiplication on points in a space of $N+1$ dimensions by representing points in $N+1$ - space in the form $A+Bu$ where A is a scalar, B is a real number, and u is a unit vector in Euclidean N-space, $\mathbf{R}^N$. The vector u belongs to the unit sphere S^{N-1} about the origin in $\mathbf{R}^N$ and is taken to have square equal to minus one: $u^2 = -1$ for all vectors in S^{N-1}. While uv is not in general defined in the higher dimensional cases, we can still study power maps of the form $z^n + k$ where k is a vector in $\mathbf{R}^{N+1}$ and $z = A+Bu$ with $u^2 = -1$ for all u in S^{N-1}. In this way we can study classes of hypercomplex iterations in arbitrary dimensions.

We are interested in the properties of iterations of functions of the form $f(z) = z^n + k$ where z is a hypercomplex variable of the form $z = A + Bu$ as described above, and k

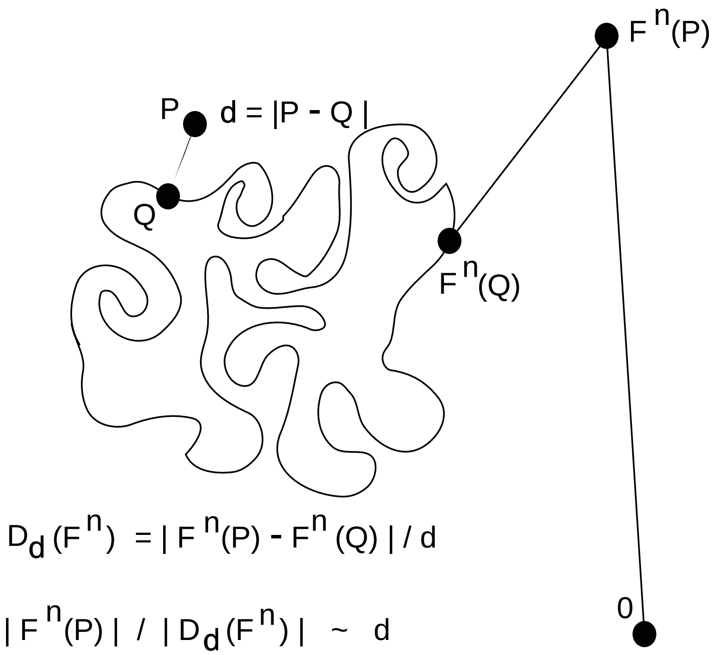
F n(P)
P
d = |P - Q |
Q
F n(Q)
Dd (F n) = | F n(P) - F n (Q) | / d
0
| F n(P) | / | Dd (F n) | ~ d

Figure 1.3: Geometry of the distance estimation for a compact fractal.

is a hypercomplex constant. In particular, we are concerned with the **Julia sets** and the **Mandelbrot sets** associated with these functions. See Section 4.4 for a more general definition of these sets than the one we are about to state. Let $f^n(z)$ denote the n-th iterate of f applied to z for n a natural number. Thus $f^3(z) = f(f(f(z)))$. The Julia set of $f(z) = z^n + k$ is the set of those points z, such that $f^n(z)$ does *not* tend to infinity as n goes to positive infinity (n=1,2,3,4,...). Let $f_k(z) = z^n + k$. The Mandelbrot set of z^n is the set of points k such that $f_k^n(0)$ does not go to infinity as n goes to infinity.

The simplest way to obtain a first look at the Julia and Mandelbrot sets is to use an inverse iteration algorithm. We will illustrate this here by explaining inverse iteration for the Julia set for $f(z) = z^2 + k$. The **inverse function** of $f(z)$ is $g(z) = \pm\sqrt{z - k}$ in the sense that $f(g(z)) = z$. By starting with a point that is not in the Julia set and iterating $g(z)$, we produce backward orbits that converge to the Julia set itself. There are some subtleties in applying this method, and we refer the reader to Chapter 8 for a discussion of some of them. Nevertheless, the inverse iteration algorithm itself is very simple. It is based on the formula

$$\sqrt{a + bu} = \pm\left(\sqrt{(1 + a)/2} + \operatorname{sign}(b)u\sqrt{(1 - a)/2)}\right),$$

valid when $a^2 + b^2 = 1$ and u is a unit pure hypercomplex number so that $u^2 = -1$. The indication sign(b) denotes the sign of the number b: that is, sign(b) is $+1$ if b is positive and -1 if b is negative. This square root formula can be applied immediately to the hypercomplex iteration $g(z) = \pm\sqrt{z^2 - k}$ by writing $z^2 - k$ in the form $\rho(a + bu)$ where ρ is a real number greater than or equal to zero, a and b are real numbers such that $a^2 + b^2 = 1$, and u is a pure hypercomplex unit. In the case of z in Euclidean three-space, we have that u lies on a unit circle in the plane and hence is determined by a choice of angle θ. As a result, the constant k for the iteration has the form $A + Bu$ where A and B are arbitrary real numbers and u is determined by an angle θ. The triple (A, B, θ) then determines the choice of a particular three dimensional "quaternionic" Julia set. Figures 1.1 and 1.2 provide examples of images produced by this inverse iteration method.

In Appendix B we include a copy of a BASIC program that can be used to compute three dimensional quaternionic Julia sets by inverse iteration. The parameters (A, B, θ) in the program are as described above.

The CD-ROM that accompanies this book exhibits many families of quaternionic Julia sets in three dimensional space. Appendix A gives the equations and parameterizations for these families. The images are produced by choosing a specific value of θ for each family, and then varying a complex constant $c = R + Ii$. Thus, a given family depends upon a 2-dimensional array of parameters (R, I). The function that is iterated for a fixed θ (and fixed $c = R + Ii$) is

$$f_\theta = e^{-i\theta} z^2 + e^{i\theta} c,$$

where z is a quaternionic variable.

The reader should note that our images come in two flavors, (A, B, θ) and (R, I, θ). The first parameterization corresponds to the point-cloud program given in Appendix B. The second parameterization is used on the CD-ROM to illustrate three-dimensional slices of four-dimensional sets, where the angle θ effects a rotation in 4-space.

Now we turn to the main topic of this book – the distance estimation algorithm. This is our new approach to distance estimation. The approach is surprisingly elementary, but it was not so easy to find!

Assume that we are given a function $F : \mathbf{R}^N \longrightarrow \mathbf{R}^N$ and let J denote the Julia set of this function. We shall say that a point z' is *outside* J (not necessarily the same as the set theoretic complement of J) if $F^n(z')$ goes to infinity as n goes to infinity. We shall assume that J is a closed and bounded subset of $\mathbf{R}^N$.

Given a point z outside J, we wish to estimate the distance δ from z to the Julia set J. Since δ is the distance from z to J, there is a point z_0 in the Julia set and a unit vector u in $\mathbf{R}^N$ such that $z_0 = z + \delta u$. For n sufficiently large, $|F^n(z)|$ will be arbitrarily large, while $|F^n(z_0)|$ is bounded by the maximal distance from the origin to points in the compact set J. See Figure 1.3. Therefore, for sufficiently large n, we have the estimate

$$|F^n(z)|/|F^n(z_0) - F^n(z)| \approx 1.$$

Letting

$$D_{\delta,u}(G) = (G(z + \delta u) - G(z))/\delta,$$

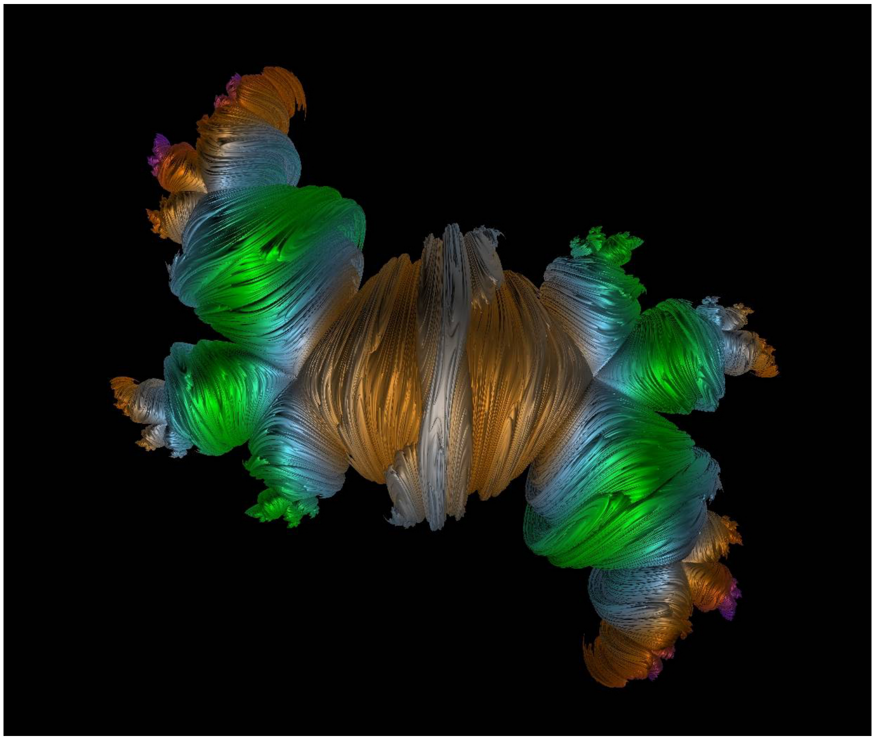

Figure 1.4: A ray-traced quaternionic fractal with parameters $\theta = 90$, $R = 0.25$, and $I = -0.05$.

we conclude that

$$|F^n(z)|/|D_{\delta,u}(F^n(z))| \approx \delta.$$

This is our basic distance estimation formula. It expresses the distance estimate in terms of the value of a discrete directional derivative. In applying this formula, we make estimates to ensure that the discrete derivative can be approximated by the exact or formal derivatives of the associated functions. See Section 7.3 for the details of this work. Our approach to distance estimation by this technique enables us to use it for hypercomplex fractals of the sort described in this introduction. Our results on distance estimation generalize and simplify the known results for iterations in the complex plane. See Chapter 3 for a review of this subtle complex analysis.

Distance estimation allows high level rendering of hypercomplex fractals such as the image shown in Figure 1.4. That is the subject of this book!

Chapter 2

Deterministic Fractals and Distance Estimation

2.1 Fractals and Visualization

Mandelbrot's fractal geometry provides both a description and a mathematical model for many of the complex forms found in nature. Shapes such as coastlines, mountains and clouds are not easily described by traditional Euclidean geometry. Nevertheless, they often possess a remarkable invariance under changes of magnification. This statistical self-similarity is an essential feature of fractals in nature. While fractals are not strictly defined by the property of statistical self-similarity [27], this concept is a useful rule-of-thumb for the presence of fractal phenomena. The simplest fractals often exhibit strict self-similarity with respect to a given scaling law. The fractals that we study in this book are more irregular in texture.

Fractals have helped reconnect research in pure mathematics with both the natural sciences and computer graphics. Within the past twenty years, fractal geometry and its concepts have become central tools in most of the natural sciences: physics, chemistry, biology, geology, meteorology and materials science, to name a few. At the same time, fractals are important to graphic designers and film makers in the creation of artificial but realistic worlds.

As fractal models become better understood through visualization, they become easier to control and study. Conversely, fractal geometry and mathematical models play a central role in the realistic rendering and modelling of natural phenomena in computer visualization.

2.2 Deterministic Fractals, Julia Sets and Mandelbrot Sets

Fractal models may be separated into two families, random and deterministic, based on their construction. The generation of both kinds of fractals may depend on streams of random numbers; the distinction is based on the influence that these random numbers have on the shape of the fractal. Altering the stream of random numbers will change the shape of a random fractal, but it will not affect the shape of a deterministic fractal.

Random fractals are used to simulate natural phenomena. The basic random fractal model is a generalization of Brownian motion called "fractal Brownian motion" [27]. More advanced methods have been developed since then in [38], [6], [34].

Historically, the first illustrations of 3D deterministic fractals were generated at coarse resolutions, conveying the basic form of the fractal. More refined techniques are costly in one way or another (memory or time). In fact, for a person just beginning to study Julia sets and Mandelbrot sets, it is a good idea to use simple coarse algorithms for a first look. It is then possible to exercise one's imagination in seeing the form of the final limit set.

The Julia and Mandelbrot sets are deterministic fractals generated by polynomial functions. There are polynomial functions whose iteration determines these fractals in point-by-point decisions. The decisions consist in testing whether a given point does or does not escape to infinity under the iteration. Inverse iteration (see Chapter 8) produces robust coarse images of deterministic fractals.

In 1982, A. Norton produced one of the first treatments of the special problems of rendering fractals. His method of generating deterministic fractals, called "boundary tracking," was an early implementation of volume visualization. Boundary tracking generates a fixed-resolution voxel approximation of a fractal set. It runs in object space, precisely $O(n^d)$ space, where d is the box-counting dimension of the object. A z-buffer was used for hidden surface removal. The gradient of the z-buffer was used to simulate tangent planes along the surface, whose normal vector was used to diffusely shade the set [29], [30].

J. Holbrook studied deterministic fractals using a brute force $O(n^3)$ computation. He also used an $O(n)$ point enumeration technique to create a point cloud approximation. The points were so sparse that he characterized them as "starfields" [18], [19].

The point enumeration technique was developed further by J. Hart, L. Kauffman, and D. Sandin. Shading and other depth cues were added to give the point cloud the appearance of a 3D surface. Unfortunately, the point clouds had large gaps that were difficult to fill in with this method [16].

Later, a ray-tracing method using distance estimation [15] was developed to efficiently render deterministic fractals. But neither a mathematical explanation nor a proof was given of the distance estimation formula beyond the complex plane.

2.3 Distance Estimation

The distance estimate is an inequality that gives a lower bound and an upper bound for the distance from an external point to the deterministic fractal itself.

Distance estimation in the complex plane was proved by [13] and has been in use for some time. An algorithm using distance estimation, called the “continuous potential method” given in [33], is an efficient algorithm for generating 2D deterministic fractals to an arbitrary level of detail. Since the exact distance from a point to a deterministic fractal is difficult to deduce, the distance estimation formula becomes essential in this algorithm.

Unlike the complex case, distance estimation in hypercomplex space appears difficult to prove by analytic means. For example, the lack of commutativity of the quaternions is problematic. As mentioned in the previous section, an efficient rendering algorithm using distance estimation was studied by Hart, Kauffman, and Sandin. These authors did not give mathematical justification for their algorithm.

It is the objective of this book to prove that these distance estimation formulas are indeed valid in hypercomplex space. In the next section we will review the analytic methods that apply to the complex plane. However, our proof of hypercomplex distance estimation, given in Chapter 7, will ultimately rely on a geometric reformulation of the problem. Our novel geometric approach to distance estimation will prove valid for both the complex plane and higher dimensions.

Part II

Classical Analysis: Complex and Quaternionic

Chapter 3

Distance Estimation in Complex Space

3.1 Complex Dynamical Systems

The dynamics of a function can be obtained by repeated application of that function to an initial starting value. For example, let f be a function and z_0 be an initial value, then the orbit of z_0 is $z_1, z_2, ..., z_n$ defined by

$$\begin{aligned} z_1 &= f(z_0), \\ z_2 &= f(z_1), \\ &\vdots \\ z_n &= f(z_{n-1}). \end{aligned}$$

A simple function with interesting properties is the quadratic

$$f(z) = z^2 + c, \tag{3.1}$$

where z is the iterated variable and c is a fixed constant.

In most cases, for a certain initial value z_0, there exists a cycle such that, for a least n, z_0 and the points $z_1, z_2, ..., z_{n-1}$ are distinct, while $z_n = z_0$. The points z_0,$z_1, z_2, ..., z_n$ are a cycle of period n. For example, the function

$$f(z) = z^2 - 1 \tag{3.2}$$

has a cycle of period 2, since $f^2(0) = 0$. All points contained in a cycle are called **periodic points**. A periodic point with period one is called a **fixed point**. Points whose orbits become cycles are called **preperiodic**. Points not in a cycle but which eventually map into a cycle are termed **strictly preperiodic**.

Cycles are classified by the eigenvalue λ of each point in the cycle. The **eigenvalue** of a periodic point is defined to be the derivative of the n-th iteration of that point, where n is the period of the point. It is easy to see that λ is the same for every point in a cycle. Once λ is determined, the cycle may then be described as

$\lambda = 0$, superattractive

$\lambda < 1$, attractive

$\lambda = 1$, indifferent

$\lambda > 1$, repelling.

The **basin of attraction** is defined as the set of all points which approach a given attracting periodic orbit. The basin of attraction $A_c(z)$ of an attractive periodic point z includes the set of points z_0 such that

$$f^k(z_0) = z \tag{3.3}$$

for some $k > 0$ and where $f(z) = z^2 + c$. An attractive cycle γ has a basin of attraction $A_c(\gamma)$. Note that for any polynomial mapping, infinity is always an attractive fixed point, and so $A_c(\infty)$ always exists [5], and

$$A_c(\infty) = \{z_0 \in \mathbf{C} \mid f^k(z_0) \to \infty \text{ as } k \to \infty\}.$$

$A_c(\infty)$ depends on c, where c is the constant in the function $f(z) = z^2 + c$.

3.2 The Quadratic Family, Julia Sets and the Mandelbrot Set

Consider the mapping $f_c\colon \mathbf{C} \to \mathbf{C}$, $f_c(z) = z^2 + c$, $c \in \mathbf{C}$. $A_c(\infty)$ has a natural boundary; that is, there are always points z_0, which generate orbits that do not approach ∞. Thus the orbits stay bounded. For example, f_c always has two fixed points, the solutions of $z^2 + c = z$. The boundary of $A_c(\infty)$ is denoted by $\partial A_c(\infty)$ and is called the **Julia Set** of f_c. We use the symbol J_c for this boundary:

$$J_c = \partial A_c(\infty).$$

The set of points

$$K_c = \mathbf{C} \setminus A_c(\infty) = \{z_0 \in \mathbf{C} \mid f_c^k(z_0) \text{ stays bounded for all } k\ \}$$

is called a **filled-in Julia set**. Obviously,

$$\partial K_c = J_c = \partial A_c(\infty).$$

That is, J_c is a frontier between orbits attracted to ∞ and orbits remaining bounded as $k \to \infty$. We will often refer to K_c and J_c as Julia sets, with the understanding that the former is a filled-in Julia set and the latter is the boundary of the set of points that approach infinity under the iteration.

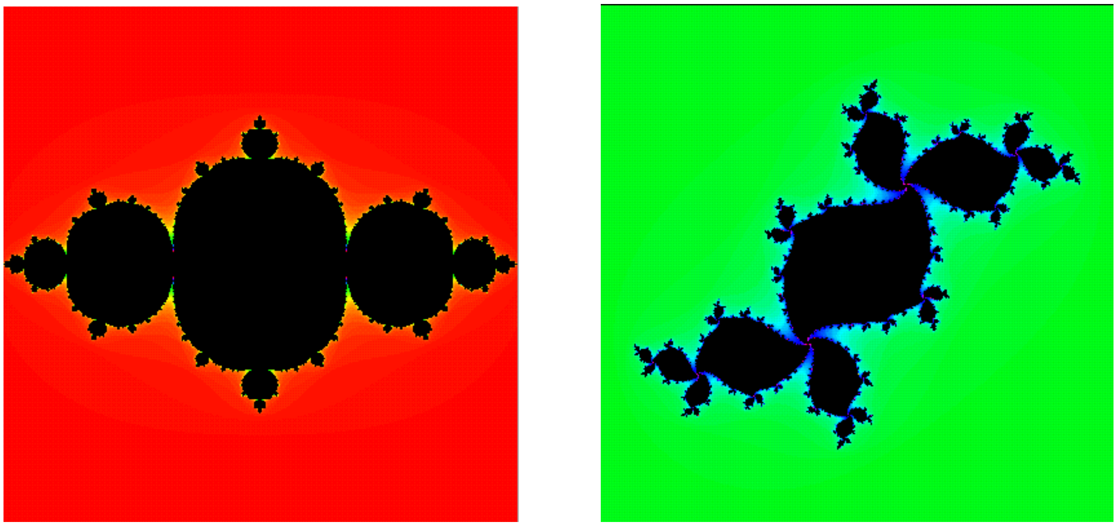

Figure 3.1: Complex Julia sets.

For most values of c, the Julia set J_c is a fractal. Moreover, the J_c's change dramatically as c changes. Figure 3.1 shows J_c's for different values of c.

Julia sets are also characterized as the closure of the set of repelling periodic points of the function being iterated:

$$J_c = \text{closure}\{\text{all repelling periodic points of } f_c\}.$$

The **Mandelbrot set** was discovered by B. B. Mandelbrot in 1980 [25]. One direct definition of the Mandelbrot set is as follows: To each point c in the complex plane, we associate the function $f_c(z) = z^2 + c$, and consider the iteration of $f_c(z)$ applied to zero. If this iteration does not go to infinity, then c is a member of the Mandelbrot set:

$$M = \{c \in \mathbf{C} \mid f_c^n(0) \not\to \infty \text{ as } n \to \infty\}.$$

It turns out that a point c is in the Mandelbrot set exactly when the corresponding Julia set K_c is connected. Thus, a classification of Julia sets in terms of connectivity can help one to understand the Mandelbrot set, and the shape of the Mandelbrot set helps one to understand the Julia sets.

It has been proved that each filled-in Julia set K_c is

- either connected
- or a Cantor set (a dust of infinitely many points).

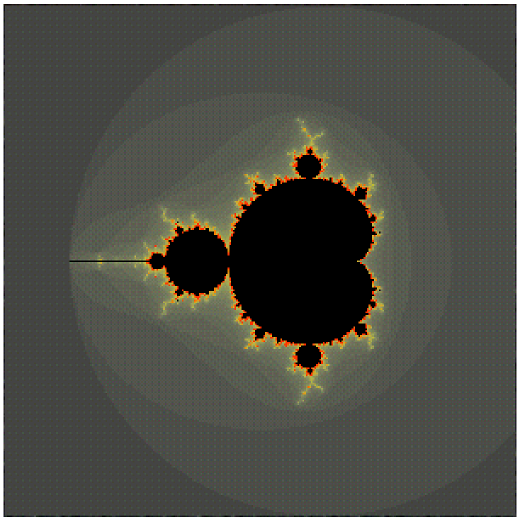

Figure 3.2: The complex Mandelbrot set.

Theorem 1 *The Mandelbrot set (as defined above) is equal to the set*

$$M' = \{c \in \mathbf{C} \mid K_c \textit{ is connected}\}.$$

In other words, M serves as a set of control parameters for K_c.

Proof: As above, Julia sets K_c can be classified into two distinct classes, connected sets or Cantor sets.

Suppose c is fixed. If we choose a very large disk of radius R, centered at the origin with boundary S_R, then K_c is inside S_R.

We now use $S_R = S_0$ as a starting set to describe an inverse iteration algorithm for computing the Julia set J_c. By examining this algorithm, we can determine whether a Julia set is a connected set or a Cantor set.

For $k = 1, 2, 3, ...$, if we compute the pre-image of S_{k-1} under f_c, i.e., with $S_0 = S_R$,

$$S_k = \{z \mid f_c(z) \in S_{k-1}\}, \tag{3.4}$$

then S_1 is a closed curve that can be thought of as a deformed circle. This is certainly true if R is very large, because f_c acts likes $f_0(z) = z^2$ near $z = \infty$, and f_c is slightly perturbed by c. Since the pre-image of S_k under f_0 is a concentric circle with radius $\sqrt{R}$, we may conclude that S_1 is indeed a connected curve. For $k > 1$, the following is a tautology

- either all S_k's are circle-like,
- or there is $k^* \geq 1$, so that S_k's are circle-like for $0 \leq k \leq k^*$ and they are not circle-like for $k \geq k^*$.

In the first case, the nested sequence of circle-like S_k's approximates K_c better and better as $k \to \infty$, and therefore, K_c must be connected and $c \in K_c$. In the second case, there are only two kinds of shapes of S_{k^*}, either two disconnected circle-like curves or two distinct self-intersecting curves. Thus it can be shown K_c is not connected and $c \notin K_c$. In other words,

$$M = \{c \mid c \in K_c\} = \{c \mid 0 \in K_c\} = \{c \mid 0 \notin A_c(\infty)\}.$$

See [33]. This completes the proof. □

3.3 The Distance Estimation Formula

Let $c \in M$ so that K_c is connected. In that case, it can be proved [13] that there is a one-to-one and onto biholomorphic mapping, with D denoting a disk of unit radius about the origin in the complex plane,

$$\phi_c \colon \mathbf{C} \setminus K_c \longrightarrow \mathbf{C} \setminus D$$

for which $\phi_c(z) \to z$ when $z \to \infty$, and which makes the following diagram commute:

$$\begin{tikzcd}
\mathbf{C} \setminus D \arrow[d, "f_0"'] & \mathbf{C} \setminus K_c \arrow[l, "\phi_c"'] \arrow[d, "f_c"] \\
\mathbf{C} \setminus D & \mathbf{C} \setminus K_c \arrow[l, "\phi_c"]
\end{tikzcd}$$

In other words, $f_0(z) = \phi_c(f_c(\phi_c^{-1}(z)))$. That is, the iteration of f_0 outside of D is equivalent to the iteration of f_c outside of K_c. A potential function for K_c is defined as

$$G(z) = \ln(|\phi_c(z)|), \tag{3.5}$$

and the escape time function is

$$\varepsilon(z) = -\ln(G_c(z)). \tag{3.6}$$

It is essential that K_c be connected, because otherwise ϕ_c does not exist. If, however, K_c is a Cantor set, that is, $c \notin \mathbf{C}$, then ϕ_c exists at least outside S_{k^*} (see Equation 3.4), and the relation $f_0(z) = \phi_c(f_c(\phi_c^{-1}(c)))$ is still true. Later we will see that we can still use distance estimation to obtain the contours of K_c, $c \notin M$.

In the complex plane, the distance estimation formulas for Julia and Mandelbrot sets are essentially the same. Here we only consider Julia sets.

Theorem 2 *Let $d(z, K_c)$ be defined as the distance from $z \in \mathbf{C}$ to the Julia set K_c. Then the distance $d(z, K_c)$ between a point z lying outside of K_c and K_c itself satisfies*

$$\frac{\sinh G(z)}{2e^{G(z)}|G'(z)|} < d(z, K_c) < \frac{2 \sinh G(z)}{|G'(z)|}, \tag{3.7}$$

where $G(z)$ is the potential at the point z.

An approximation of Inequality 3.7 is

$$\frac{|z_n|}{2|z_n|^{\frac{1}{2^n}}|z_n'|}\ln|z_n| < d(z,K_c) < \frac{|z_n|}{|z_n'|}\ln|z_n|. \tag{3.8}$$

In next few sections, we will prove the above distance estimation formula and this approximation. It is the approximation that has the most practical value for making images of Julia sets, and it is the approximation that generalizes to quaternion and higher dimensional fractals, as we will see in Section 7.3.

3.4 Schwarz's Lemma and an Upper Bound of the Distance Estimate

We will use some results from complex analysis, such as Schwarz's Lemma, to prove the upper bound for the distance estimate in the complex numbers. First we have the following lemmas.

Lemma 3 (Parseval's Identity) *Assume that f has the Taylor expansion*

$$f(z)=\sum_{n=0}^{\infty}a_n(z-z_0)^n,$$

valid in $N(z_0;R)=\{z \mid |z-z_0|\le R\}$. If $0\le r\le R$, then

$$\frac{1}{2\pi}\int_0^{2\pi}|f(z_0+re^{i\theta})|^2d\theta=\sum_{n=0}^{\infty}|a_n|^2r^{2n}.$$

Proof: Since

$$\int_0^{2\pi}|f(z_0+re^{i\theta})|^2d\theta=\int_0^{2\pi}\left|\sum_{n=0}^{\infty}a_n(z_0+re^{i\theta}-z_0)^n\right|^2d\theta=\int_0^{2\pi}\left|\sum_{n=0}^{\infty}a_n(re^{i\theta})^n\right|^2d\theta$$

$$=\int_0^{2\pi}\left|\sum_{n=0}^{\infty}a_n(r(\cos(n\theta)+i\sin(n\theta)))^n\right|^2d\theta,$$

and because for any integers m and n,

$$\int_0^{2\pi}\cos(m-n)d\theta=0,$$

where $m\neq n$, we have,

$$\int_0^{2\pi}\left|\sum_{n=0}^{\infty}a_n(r(\cos(n\theta)+i\sin(n\theta)))^n\right|^2d\theta=\int_0^{2\pi}\left|\sum_{n=0}^{\infty}a_nr^n\cos(n\theta)+i\sum_{n=0}^{\infty}a_nr^n\sin(n\theta)\right|^2d\theta$$

$$=\sum_{n=0}^{\infty}|a_n^2r^{2n}|=\sum_{n=0}^{\infty}|a_n|^2r^{2n}.$$

Hence, we have proved

$$\frac{1}{2\pi}\int_0^{2\pi}|f(z_0+re^{i\theta})|^2d\theta=\sum_{n=0}^{\infty}|a_n|^2r^{2n}.$$

□

Lemma 4 *If $M(r)$ is the maximum of $|f|$ on the circle $|z-z_0|=r$, then*

$$\sum_{n=0}^{\infty}|a_n|^2r^{2n}\leq M^2(r).$$

Proof: Since

$$\frac{1}{2\pi}\int_0^{2\pi}|f(z_0+re^{i\theta})|^2d\theta\leq\frac{1}{2\pi}M^2(r)\int_0^{2\pi}d\theta=M^2(r),$$

by Parseval's Identity,

$$\sum_{n=0}^{\infty}|a_n|^2r^{2n}\leq M^2(r).$$

□

Lemma 5 (Maximum-Modulus Theorem) *Assume that f is analytic on an open region S in $\mathbf{C}$, and suppose f is not a constant. Let τ be a rectifiable Jordan curve such that both τ and its inner region lie within S. If $|f(z)|\leq M$ on τ, then*

$$|f(z)|<M$$

at all points inside τ.

Proof: Suppose to the contrary that f is a constant and z_0 is an inner point such that

$$f(z_0)=M,$$

then

$$f(z_0)=\sum_{n=0}^{\infty}a_n(z-z_0)^n=a_0.$$

By the previous lemma,

$$\sum_{n=0}^{\infty}|a_n|^2r^{2n}\leq M^2(r)=|f(z_0)|^2=a_0^2,$$

so we have $a_n=0$ for $n=1,2,....$ Then $f(z)$ is constant, a contradiction! □

Now we can prove Schwarz's Lemma.

Lemma 6 (Schwarz's Lemma) *Let f be analytic on the neighborhood $N(0;1)$. Suppose that $f(0)=0$ and $f(z)\leq 1$. If $|z|<1$, then $|f'(0)|\leq 1$ and $|f(z)|\leq|z|$.*

Proof: For any z_0 such that $|z_0| < 1$, we can define a disk $|z| < 1 - \varepsilon$, for certain $0 < \varepsilon < 1$, such that z_0 is inside the disk. Let $g(z) = \frac{f(z)}{z}$ and $g(0) = f'(0)$. For $|z| = 1 - \varepsilon$ we have $|g(z)| < \frac{1}{1-\varepsilon}$, and by the Maximum-Modulus Theorem $|g(z_0)| < \frac{1}{1-\varepsilon}$. We can choose any ε smaller than the given ε, and therefore $|g(z_0)| \leq 1$ for any z_0, $|z_0| < 1$. That is, $|f(z_0)| \leq |z_0|$. So, $|f(z)| \leq |z|$, for $|z| < 1$, and $|f'(0)| = |g(0)| \leq 1$. □

The following lemma, which we will use in the proof of the upper bound of the distance estimate, is an extension of Schwarz's Lemma. Since it can be obtained directly from the previous lemma, we omit the proof.

Lemma 7 *Let f be analytic on neighborhood $N(a; r)$ with $r > 0$. Suppose that $f(a) = 0$ and $f(z) \leq 1$. If $|z - a| < r$, then $|f(z)| \leq \frac{|z|}{r}$ and $|f'(a)| \leq \frac{1}{r}$.*

Theorem 8 *The distance from a point z, $z \in \mathbf{C}$, $z \notin K_c$ to the Julia set K_c is strictly less than $\frac{2\sinh G(z)}{|G'(z)|}$, where $G(z)$ is the potential of the point z. That is,*

$$d(z, K_c) < \frac{2\sinh G(z)}{|G'(z)|}.$$

Proof: Let ϕ_c be the one-to-one and onto biholomorphic mapping

$$\phi_c \colon \mathbf{C} \setminus K_c \longrightarrow \mathbf{C} \setminus D$$

as defined in the previous section, where $D = \{z \mid |z| < 1\}$. Let

$$W(\mu) = \frac{1}{\phi_c(\mu)},$$

and

$$G(\mu) = -\ln |W(\mu)|.$$

Clearly, we can see $G(\mu) = \ln |\phi_c(\mu)|$. We define $F(w)$ as follows,

$$F(w) = \frac{w - w_0}{1 - w\overline{w}_0},$$

where $w_0 = W(\mu_0)$, μ_0 is in the complement of K_c, i.e. $\mu \in \mathbf{C} \setminus K_c$.

First, we are going to prove

$$F'(w_0) = \frac{1}{1 - w_0\overline{w}_0} > 0.$$

It is not difficult to see that

$$F'(w_0) = \frac{1}{1 - w_0\overline{w}_0}$$

and

$$|W(\mu_0)| = \frac{1}{|\phi_c(\mu_0)|}.$$

By the definition of $\phi(z)$ and $\mu \notin K_c$,

$$|W(\mu_0)| < 1.$$

Hence,

$$W(\mu_0)\overline{W(\mu_0)} < 1$$

and

$$\frac{1}{1 - w_0\overline{w}_0} > 0,$$

where $w_0 = W(\mu_0)$.

Second, we prove: $F \circ W$ maps the disk of radius r about μ_0 to the unit disk with $F(W(\mu_0)) = 0$, where $r > 0$ such that the open disk $N(\mu_0; r) \notin K_c$.

In order to prove this statement, we need to verify first that $F(w)$ is the fractional linear transformation of the unit disk which takes w_0 to 0.

We can show, if $|w| \leq 1$, then $|F(w)| \leq 1$. Also if $|w| < 1$, then $|F(w)| < 1$.

Let $w = x + iy$, $w_0 = a + bi$. Then

$$x^2 + y^2 < 1$$

and

$$\begin{aligned} |(w - w_0)|^2 &= |(x + iy) - (a + ib)|^2 \\ &= (x^2 + y^2) + (a^2 + b^2) - 2ax - 2by, \\ |1 - w\overline{w}_0|^2 &= |1 - (x + iy)(a + ib)|^2 \\ &= 1 + (x^2 + y^2)(a^2 + b^2) - 2ax - 2by. \end{aligned}$$

Since

$$(x^2 + y^2)(1 - a^2 - b^2) < (1 - a^2 - b^2),$$

we have

$$|(w - w_0)|^2 < |1 - w\overline{w}_0|^2.$$

That is,

$$|(w - w_0)| < |1 - w\overline{w}_0|.$$

So

$$|F(w)| < 1.$$

If $|w| = 1$, then we can show $|F(w)| = 1$. Since if $|w| = 1$, then $w^{-1} = \overline{w}$, and we have

$$|F(w)| = \frac{|w - w_0|}{|1 - w\overline{w}_0|} = \frac{|w||1 - w_{-1}w_0|}{|1 - w\overline{w}_0|} = \frac{|1 - \overline{w}w_0|}{|1 - w\overline{w}_0|} = 1.$$

We also need to show that $W(\mu) = \frac{1}{\phi(\mu)}$ maps the disk $N(\mu_0; r)$ of radius r about μ_0 to the unit disk D.

Actually, ϕ_c maps $\mathbf{C} \setminus K_c$ to $\mathbf{C} \setminus D$, and $h(z) = \frac{1}{z}$ maps $\mathbf{C} \setminus D$ to unit disk D. Hence, $W(\mu)$ maps $\mathbf{C} \setminus K_c$ to unit disk D, that is,

$$\phi_c \colon \mathbf{C} \setminus K_c \longrightarrow \mathbf{C} \setminus D,$$

$$h(z) = \frac{1}{z} \colon \mathbf{C} \setminus D \longrightarrow D,$$

$$W \colon \mathbf{C} \setminus D \longrightarrow D.$$

So, W maps the disk $N(\mu_0; r)$ of radius r to unit disk D, and $F \circ W$ maps the disk of radius r about μ_0 to the unit disk with $F(W(\mu_0)) = 0$.

Now we can prove the upper bound for the distance estimate. Let r be the distance from μ_0 to the Julia set K_c. For any μ such that $|\mu - \mu_0| < r$, we have $|F(W(\mu))| \leq 1$, and $F(W(\mu_0)) = 0$. By Schwarz's Lemma,

$$|F'(W(\mu_0))W'(\mu_0)| < \frac{1}{r}.$$

That is,

$$r < \frac{1}{|F'(W(\mu_0))W'(\mu_0)|} = \frac{|1 - w_0\overline{w}_0|}{|W'(\mu_0)|}.$$

Since

$$|w_0| = |w(\mu_0)| = e^{-G(\mu_0)},$$

and

$$1 - w_0\overline{w}_0 = 1 - |w_0|^2 = 1 - (e^{-G(\mu_0)})^2 = 1 - e^{-2G(\mu_0)} = e^{-G(\mu_0)} 2 \sinh G(\mu_0),$$

and

$$|W'(\mu_0)| = |(e^{-G(\mu_0)})'| = e^{-G(\mu_0)} |G'(\mu_0)|,$$

we have

$$r < \frac{e^{-G(\mu_0)} 2 \sinh G(\mu_0)}{e^{-G(\mu_0)} |G'(\mu_0)|} = \frac{2 \sinh G(\mu_0)}{|G'(\mu_0)|}.$$

So for any point z, $z \in \mathbf{C}$, $z \notin K_c$, the distance from z to the Julia set K_c is strictly less than $\frac{2 \sinh G(z)}{|G'(z)|}$. □

3.5 The Koebe 1/4 Theorem and a Lower Bound for the Distance Estimate

We will now prove a theorem in complex analysis called the Koebe 1/4 theorem, which allows us to obtain a lower bound for the distance estimate. To prove this theorem, we first need some definitions and some lemmas.

Definition 1 (univalent) *If an analytic function is one-to-one, then it is univalent.*

Definition 2 **(class $\sum_0$)** *An analytic function f is in class $\sum_0$ if and only if f is univalent in $D = \{z \mid |z| < 1\}$, and f can be normalized so that the Taylor expansion at 0 has the form*

$$f(z) = z + \sum_{n=2}^{\infty} a_n z^n. \tag{3.9}$$

Definition 3 **(class $\sum_\infty$)** *An analytic function f is in class $\sum_\infty$ if and only if f is univalent in $D^0 = \{z \mid |z| > 1\}$, and $f$can be normalized so that the Taylor expansion at 0 has the form*

$$f(z) = z + \sum_{n=0}^{\infty} b_{-n} z^{-n}. \tag{3.10}$$

Lemma 9 *If $f \in \sum_0$ and w is in the complement of $f(D)$, then the function*

$$h(z) = \frac{f(z)}{1 - w^{-1} f(z)}$$

is in $\sum_0$.

Proof: Since $f(z) \neq w$ for $z \in D$, $h(z)$ is an analytic function. Since the Möbius transformation

$$M(u) = \frac{u}{1 - w^{-1} u}$$

is a one-to-one function, $h(z)$ is univalent. It is obvious that $h(z)$ has the form

$$h(z) = z + \sum_{n=2}^{\infty} a_n z^n.$$

So $h(z) \in \sum_0$. □

Lemma 10 *For any function $g(z) \in \sum_\infty$ given by Equation 3.10,*

$$a \sum_{n=1}^{\infty} n |b_{-n}|^2 \leq 1. \tag{3.11}$$

Proof: For $\rho > 1$, let τ_ρ denote the curve $\omega = g(\rho e^{i\tau})$, $0 \leq \tau \leq 2\pi$. Because g is univalent, τ_ρ is a positively oriented Jordan curve. Then

$$\alpha(\rho) = \frac{1}{2i} \int_{\tau_\rho} \overline{\omega} d\omega = \frac{1}{2} \int_{\tau_\rho} v du - u dv$$

is an area enclosed by τ_ρ. Hence $\alpha(\rho) \geq 0$. On the other hand, explicit calculation yields

$$\alpha(\rho) = \frac{1}{2} \int_0^{2\pi} \left(\rho e^{-i\tau} + \sum_{n=0}^{\infty} \overline{b_{-n}} \rho^{-n} e^{-in\tau} \right) \left(1 - \sum_{n=1}^{\infty} n b_{-n} \rho^{-n-1} e^{-i(n+1)\tau} \right) \rho e^{i\tau} d\tau.$$

All cross terms can be cancelled, so we have

$$\alpha(\rho) = \pi \left(\rho^2 - \sum_{n=1}^{\infty} n|b_{-n}|^2 \rho^{-2n} \right).$$

Let $\rho \to 1$, then

$$\sum_{n=1}^{\infty} n|b_{-n}|^2 \leq 1. \tag{3.12}$$

Note, from Equation 3.11,

$$|b_{-n}| \leq \frac{1}{\sqrt{n}}, \text{ for } n = 1, 2, 3, \ldots$$

Especially for $n = 1$, we have

$$|b_{-1}| \leq 1. \tag{3.13}$$

□

Lemma 11 *If $f \in \sum_0$, then the function*

$$g(z) = \frac{1}{f\left(\frac{1}{z}\right)} + \omega_0$$

is in $\sum_\infty$ for any ω_0.

Proof: Clearly, g is analytic in D^o. If $g(z_1) = g(z_2)$, then $f(z_1^{-1}) = f(z_2^{-2})$, $z_1 = z_2$. So $g(z)$ is univalent, and it is easy to see $g(z)$ can be normalized such that it has the form of Equation 3.10. That is, $g(z) \in \sum_\infty$. □

Lemma 12 *If $f \in \sum_0$, then the function*

$$h(z) = [f(z^2)]^{\frac{1}{2}} = zs(z^2)$$

is in $\sum_0$, where $s(z) = [z^{-1}f(z)]^{\frac{1}{2}}$.

Proof: Clearly, h is analytic in D^o. If $h(z_1) = h(z_2)$, then $s(z_1^2) = s(z_2^2)$, and since $h(z) = zs(z^2)$, it follows that $z_1 = z_2$. So $h(z)$ is univalent, and $h(z) \in \sum_0$. □

Lemma 13 *Let $f \in \sum_0$ be given by Equation 3.9. Then $|a_2| \leq 2$.*

Proof: By Lemma 12, the function

$$h(z) = [f(z^2)]^{\frac{1}{2}} = z\left(1 + \frac{1}{2}a_2 z^2 + \cdots\right)$$

is in $\sum_0$. And by Lemma 11, the function

$$g(z) = [f(z^2)]^{-\frac{1}{2}} = z\left(1 - \frac{1}{2}a_2 z^{-2} + \cdots\right)$$

is in $\sum_\infty$. So $|\frac{1}{2}a_2| \leq 1$, that is,

$$|a_2| \leq 2$$

□

Theorem 14 (Koebe $\frac{1}{4}$ Theorem) *Let $f \in \sum_0$. Then $f(D)$ contains the disk $|\omega| < \frac{1}{4}$.*

Proof: Let $\omega \in \mathbf{C} \setminus f(D)$. By Lemma 9, the function

$$h(z) = \frac{f(z)}{1 - \omega_1 f(z)}$$

is in $\sum_0$. The Taylor series of h at 0 is

$$h(z) = \left(z + a_2 z^2 + \cdots\right)\left(1 + \frac{1}{\omega} z + \cdots\right)$$

$$= z + \left(a_2 + \frac{1}{\omega}\right) z^2 + \cdots.$$

By Lemma 13,

$$\left|a_2 + \frac{1}{\omega}\right| \leq 2.$$

But also $|a_2| \leq 2$. Therefore,

$$|\omega^{-1}| \leq 4 \text{ or } |\omega| \geq \frac{1}{4}.$$

If $f : D \longrightarrow \mathbf{C}$ is an analytic function from the open disk $D = \{z \mid |z| < 1\}$ into the complex plane, and if f is one-to-one with $f(0) = 0$ and $|f'(0)| = 1$, then $f(z) \in \sum_0$ and $f(D) \supset D\left(\frac{1}{4}, 0\right)$. If $F : D \longrightarrow \mathbf{C}$ is an analytic function from the open disk $D = \{z \mid |z| < 1\}$ into the complex plane, and if F is one-to-one with $F(0) = a$ and $|F'(0)| = b$, then $F(z)$ can be normalized to $f(z)$ such that $f(z) \in \sum_0$. Therefore, $f(D) \supset D\left(\frac{1}{4}, 0\right)$ and $F(D) \supset D\left(\frac{b}{4}, a\right)$. □

Theorem 15 *The distance from a point z, $z \in \mathbf{C}$, $z \notin K_c$ to the Julia set K_c is strictly greater than $\frac{\sinh G(z)}{2e^{G(z)}|G'(z)|}$. That is,*

$$d(z, K_c) > \frac{\sinh G(z)}{2e^{G(z)}|G'(z)|}.$$

Proof: Let ϕ_c be the one-to-one and onto biholomorphic mapping

$$\phi_c \colon \mathbf{C} \setminus K_c \longrightarrow \mathbf{C} \setminus \overline{D}$$

as defined in the previous section, where $\overline{D} = \{z \mid |z| \leq 1\}$.
Notation: For the rest of this proof, we let ϕ denote ϕ_c.
Let

$$r(z) = \frac{1}{z}.$$

Then

$$r \circ \phi \colon \mathbf{C} \setminus K_c \longrightarrow D \setminus \{0\}$$

and

$$(r \circ \phi)^{-1} \colon D \setminus \{0\} \longrightarrow \mathbf{C} \setminus K_c.$$

For any $c_0 \in \mathbf{C} \setminus K_c$, let

$$d = r(\phi(c_0)) = \frac{1}{\phi(c_0)} \quad \in D \setminus \{0\}.$$

Let the Möbius Transformation be $m_d(z)$,

$$m_d(z) = \frac{z - d}{1 - \overline{d}z}.$$

Then $m_d(z)$ transforms unit disk to unit disk and it maps d to 0 and 0 to $-d$. Define ψ as

$$\psi = (m_d \circ r \circ \phi)^{-1}(|d|z).$$

So ψ is a univalent function. Since

$$M_d \circ r \circ \phi \colon \mathbf{C} \setminus K_c \longrightarrow D \setminus \{-d\},$$

we have

$$\psi \colon D \longrightarrow \mathbf{C} \setminus K_c,$$

and ψ maps 0 to c_0, that is

$$\psi(0) = c_0. \tag{3.14}$$

We can prove:

$$|\psi'(0)| = R = \frac{|\phi^2(c_0)| - 1}{|\phi(c_0)||\phi'(c_0)|}.$$

Since

$$m_d \circ r \circ \phi(z) = \frac{\frac{1}{\phi(z)} - d}{1 - \overline{d}\frac{1}{\phi(z)}} = \frac{1 - d\phi(z)}{\phi(z) - \overline{d}},$$

we have

$$(m_d \circ r \circ \phi(z))^{-1} = \left(\frac{1 - d\phi(z)}{\phi(z) - \overline{d}}\right)^{-1},$$

and

$$\begin{aligned} \frac{d(m_d \circ r \circ \phi(z))}{dz} &= \frac{-d\phi'(z)(\phi(z) - \overline{d}) - \phi'(z)(1 - d\phi(z))}{(\phi(z) - \overline{d})^2} \\ &= \frac{\phi'(z)(-d\phi(z) + |d|^2 - 1 + d\phi(z))}{(\phi(z) - \overline{d})^2} \\ &= \frac{\phi'(z)(|d|^2 - 1)}{(\phi(z) - \overline{d})^2}. \end{aligned}$$

Hence,

$$\left.\frac{d(m_d \circ r \circ \phi(z))}{dz}\right|_{c_0} = \frac{\phi'(c_0)(|d|^2 - 1)}{(\phi(c_0) - \overline{d})^2}$$

$$= \frac{\phi'(c_0)(|d|^2 - 1)}{(\frac{1}{d} - \overline{d})^2} = \frac{\phi'(c_0)d^2}{|d|^2 - 1}.$$

So

$$\left|\frac{d(\psi(z))}{dz}\right| = \left|\frac{d(m_d \circ r \circ \phi(z))^{-1}}{dz}\right| = \frac{1}{\frac{|\phi'(c_0)||d|^2}{|d|^2-1}}|d|, \text{ (note: } z = |d|c_0)$$

$$= \frac{|d|^2 - 1}{|\phi'(c_0)||d|} = \frac{|\phi(c_0)|^2 - 1}{|\phi'(c_0)||\phi(c_0)|},$$

that is,

$$|\psi'(0)| = R = \frac{|\phi^2(c_0)| - 1}{|\phi(c_0)||\phi'(c_0)|}. \tag{3.15}$$

Then by Equation 3.14, Equation 3.15 and the Koebe $\frac{1}{4}$ Theorem,

$$D\left(\frac{R}{4}, c_0\right) \subset \psi(D).$$

So the distance from c_0 to K_c is at least $R/4$. By the definition of $G(c_0) = \ln(|\phi(c_0)|)$, and letting $z = c_0$,

$$d(z, K_c) > \frac{R}{4} = \frac{\sinh G(z)}{2e^{G(z)}|G'(z)|}.$$

□

3.6 An Approximation of the Distance Estimation Formula

We will use distance estimation as a technique for generating images of Julia and Mandelbrot sets. In order to implement this technique, it is best to have an easily computable distance estimation formula. To this end, we now give an approximation to the distance estimate we proved using the potential function. The approximation uses only derivatives of the function being iterated. It is this approximation formula that will be generalized to hypercomplex fractal iterations in Section 7.3.

Theorem 16 *The distance estimation formula:*

$$\frac{\sinh G(z)}{2e^{G(z)}|G'(z)|} < d(z, K_c) < \frac{2\sinh G(z)}{|G'(z)|} \tag{3.16}$$

can be approximated as

$$\frac{|z_n|}{2|z_n|^{\frac{1}{2^n}}|z_n'|} \ln|z_n| < d(z, K_c) < \frac{|z_n|}{|z_n'|} \ln|z_n|. \tag{3.17}$$

Proof: The theorem is true in greater generality (see Section 7.3), but here we give the full proof for the case where $z_{n+1} = z_n^2 + c$ over the complex numbers. Let

$$d_- = \frac{|z_n|}{2|z_n|^{\frac{1}{2^n}}|z_n'|} \ln |z_n|$$

and

$$d_+ = \frac{|z_n|}{|z_n'|} \ln |z_n|.$$

Then we only need to prove that

$$d_- = \frac{\sinh G(z)}{2e^{G(z)}|G'(z)|}$$

and

$$d_+ = \frac{2 \sinh G(z)}{|G'(z)|},$$

when $z \to \infty$. Let ϕ_c be the one-to-one and onto biholomorphic mapping

$$\phi_c \colon \mathbf{C} \setminus K_c \longrightarrow \mathbf{C} \setminus \overline{D}$$

as defined in Section 3.3:

$$\begin{tikzcd}
\mathbf{C} \setminus D \arrow[d, "f_0"'] & \mathbf{C} \setminus K_c \arrow[l, "\phi_c"'] \arrow[d, "f_c"] \\
\mathbf{C} \setminus D & \mathbf{C} \setminus K_c. \arrow[l, "\phi_c"]
\end{tikzcd}$$

Let

$$f_c(z) = z^2 + c,$$

and

$$f_0(z) = z^2.$$

When $z \to \infty$,

$$f_0(z) \approx \phi_0(f_c(\phi_c^{-1}(z))).$$

Thus,

$$(\phi_c(z))^2 \approx \phi_c(f_c(z)).$$

As defined in Equation 3.5, $G(z) = \ln |\phi_c(z)|$, so when $z \to \infty$,

$$G(f(z)) = \ln |\phi_c(f_c(z))| \approx \ln(|\phi_c(z)|^2) = 2 \ln |\phi_c(z)|,$$

and

$$G(z) \approx \frac{1}{2} G(f_c(z)).$$

So

$$G(z) \approx \frac{1}{2^n} G(f_c^n(z)) = \frac{1}{2^n} \ln|\phi_c(f_c^n(z))|.$$

When $n \to \infty$, we have $f_c^n(z) \to \infty$ and $\phi_c(f_c^n(z)) = \phi_c(z_n) \to z_n$, where $z_n = f_c^n(z_0)$. So

$$G(z) = \frac{\ln|z_n|}{2^n}.$$

Then

$$G'(z) = \frac{z_n'}{2^n|z_n|},$$

and

$$e^{G(z)} = e^{\frac{\ln|z_n|}{2^n}} = |z_n|^{\frac{1}{2^n}}.$$

Note when $z \to 0$, we have $\sinh(z) \approx \frac{z}{2}$, and when $n \to \infty$, we have $G(z) = \frac{\ln|z_n|}{2^n} \to 0$. So

$$\frac{2\sinh G(z)}{|G'(z)|} \approx \frac{|G(z)|}{|G'(z)|} = \frac{\ln|z_n|2^n}{2^n(|z_n'|/|z_n|)}$$

$$= \frac{|z_n|}{|z_n'|} \ln|z_n| = d_+$$

and

$$\frac{|\sinh G(z)|}{|G'(z)e^{G(z)}|} \approx \frac{|G(z)|}{2|G'(z)e^{G(z)}|} \approx \frac{|z_n|\ln|z_n|}{2|z_n|^{\frac{1}{2^n}}|z_n'|} = d_-$$

□

Chapter 4

Quaternion Analysis

We can use the tools of hypercomplex analysis to discuss hypercomplex deterministic fractals. The first hypercomplex space is the quaternion space. Quaternions were discovered in 1843 by Irish physicist and mathematician William R. Hamilton [14]. Attempting to define a three-dimensional multiplication, he found it is necessary instead to extend to four dimensions. After this discovery, Hamilton and his contemporaries devoted considerable effort to advocating the application of quaternions to physics and other disciplines.

We can describe quaternions as an extension of the complex plane, comparable to the view of complex numbers as an extension of the real line. Complex numbers provide an extension of the concept of "number," which permits us to consider numbers as two-dimensional quantities. In other words, complex numbers are a set of rules for multiplying and adding points in two dimensions. Similarly, quaternions may be regarded as a way of extending the notion of number to four dimensions: the rules for quaternion multiplication and addition provide a way to do arithmetic with four-dimensional quantities.

In this chapter, we give an introduction to quaternion analysis and a discussion of regular functions. We show that the mathematical procedures used to study complex Julia and Mandelbrot sets are not adaptable to the quaternionic case, due to the lack of commutativity of the quaternions. We then discuss the difficulties involved in proving the analogous distance estimation formula in quaternions by analytic methods. While this is a negative result, we do believe that these basics of quaternion analysis will eventually prove useful in the subject of quaternion iterations. In fact, it was this negative result that propelled us to find the more geometric approach to distance estimation described in Chapter 7.

4.1 The Quaternions

We denote the four-dimensional real associative algebra of the quaternions by $\mathbf{H}$. The quaternions can be seen as a direct sum $\mathbf{H} = \mathbf{R} \oplus \mathbf{P}$, where $\mathbf{R}$ is the real number space, which is embedded in $\mathbf{H}$ by identifying $1 \in \mathbf{R}$ with the identity $1 \in \mathbf{H}$, and $\mathbf{P}$ is an oriented three-dimensional Euclidean vector space. A typical quaternion value,

$$q = t + ix + jy + kz,$$

is a four-tuple of independent real values (t, x, y, z) assigned to one real axis and three imaginary axes i, j, k such that

$$i^2 = j^2 = k^2 = -1, \tag{4.1}$$

$$ij = k;\ jk = i;\ ki = j, \tag{4.2}$$

$$ji = -k;\ kj = -i;\ ik = -j. \tag{4.3}$$

We note that the multiplication rule for quaternions is noncommutative. For example, $ij = -ji$. The result of multiplying two quaternions depends on their order.

There are several properties and notations of quaternion Julia sets that are used in this discussion. We state them as follows,

$$\overline{q} = t - ix - jy - kz, \tag{4.4}$$

$$|q| = \sqrt{q\overline{q}} = \sqrt{(t^2 + x^2 + y^2 + z^2)} \quad \in \mathbf{R} \tag{4.5}$$

$$\mathrm{Re}(q) = \frac{1}{2}(q + \overline{q}) = t \tag{4.6}$$

$$\mathrm{Pu}(q) = \frac{1}{2}(q - \overline{q}) = ix + jy + kz \quad \in \mathbf{P} \tag{4.7}$$

$$Un(q) = \frac{q}{|q|} \quad \in S, \tag{4.8}$$

where S is the unit sphere in $\mathbf{H}$. The inverse of q is defined as

$$q^{-1} = \frac{\overline{q}}{|q|^2}.$$

The inner product is

$$< q_1, q_2 >= \mathrm{Re}(q_1\overline{q_2}) = t_1t_2 + x_1x_2 + y_1y_2 + z_1z_2. \tag{4.9}$$

Note that if u_1 and u_2 are unit quaternions, that is, $|u_1| = |u_2| = 1$, the map $q \longrightarrow u_1qu_2$ is orthogonal with respect to the inner product as defined in Equation 4.9 and has determinant one. Conversely, any rotation of $\mathbf{H}$ is of the form $q \longrightarrow u_1qu_2$ for some $u_1, u_2 \in \mathbf{H}$.

4.2 Rotations of 3-Space

One of the most intriguing properties of Hamilton's quaternions is their ability to represent rotations of three-dimensional space. In order to explain this, we represent 3-space, $\mathbf{R}^3$, as the set of pure quaternions of the form $Z = ai + bj + ck$ where a, b and c are real numbers. If g is a unit quaternion as described in the previous section, then the mapping $\rho : \mathbf{R}^3 \longrightarrow \mathbf{R}^3$ defined by the equation $\rho(Z) = gZg^{-1}$ describes a rotation of 3-space by angle θ around the axis u when

$$g = cos(\theta/2) + sin(\theta/2)u.$$

Here, u is a unit length pure quaternion, hence a vector direction in 3-space. Any unit quaternion can be represented in this form.

We leave the proof of this result as an exercise for the reader, with the admonition to try it out with a specific axis, such as k. For example, if $g = k$, then $g = cos(\pi/2)+sin(\pi/2)k$, and it is easy to see that conjugation by k induces a 180 degree rotation around the k-axis.

This method of representing rotations using quaternions is actually quite useful in computer graphics for performing rotations without using matrices.

A useful consequence of this result about rotations is the fact that, given any quaternion q, there is a unit quaternion p such that $p^{-1}qp$ is of the form $a + bi$, for some real numbers a and b. In other words, any quaternion can be rotated into a complex number by conjugation with the appropriate unit quaternion.

Another useful consequence is the fact that one can determine the axis of a composition of rotations from the individual rotations just by multiplying the corresponding quaternions. Once again, we leave this as an exploration for the reader, and suggest that she think about the result of rotating a cube by 90 degrees around one axis, followed by a rotation of 90 degrees around a different axis.

4.3 Quaternion Polynomials

Note that if we identify the quaternion i with the complex number i, the complex numbers can be regarded as a subset of the quaternions, and complex polynomials can be regarded as polynomials over quaternions as well. An expression like

$$ax^2 + bx + c$$

with $a, b, c \in \mathbf{C}$ is a quaternion polynomial as well. However, the noncommutativity of the quaternions implies that many polynomials cannot be so simply described. For example, unless the coefficients a and b are real, the above polynomial is not equal to

$$x^2a + bx + c$$

nor

$$x^2a + xb + c$$

nor

$$xax + bx + c,$$

etc.

A quaternion polynomial can be defined as

$$p(z) = \sum_{k=0}^{l} \sum_{i=0}^{m} p_{i0} z p_{i1} z ... z p_{ik}, \tag{4.10}$$

where l and m are integers, l is the degree of the polynomial, and $p_{ij} \in \mathbf{H}$ are quaternions. $i = 0, 1, 2, ..., m$, $j = 0, 1, ..., k$.

For the purpose of this discussion, we consider only a subset of such polynomials of degree 2, for instance,

$$p(z) = p_2 z^2 q_2 + p_1 z q_1 + q_3.$$

We shall see that even these polynomials introduce a wealth of structure not seen in the complex mappings.

4.4 Quaternion Julia Sets and Mandelbrot Sets

A Julia set is generated by a quaternion polynomial. We know that a quaternion polynomial can be used to define a dynamical system on the quaternions: If $p(z)$ is a quaternion polynomial, then for any quaternion q, $p(q)$ is another quaternion. With the aid of computers, quaternions can easily be iterated to evaluate long-term behavior. We can still speak of attractive and repulsive cycles, basins of attraction and the like, where the notion of complex numbers is replaced by quaternions.

We can generalize the definition of Julia sets for arbitrary polynomials: The Julia set of a polynomial $p(z)$ is the boundary of the set of quaternions q such that $|p^n(q)|$ converges to infinity as n becomes large.

Let us denote by $A_p(\infty)$ the basin of attraction of the Julia set generated by the polynomial $p(z)$. Then the **Julia set** of $p(z)$ is defined as

$$J_p = \partial A_p(\infty),$$

and

$$K_p = \mathbf{C} \setminus A_p(\infty) = \{z_0 \in \mathbf{H} \mid p^k(z_0) \text{ stays bounded for all } k\}$$

is called a **filled-in Julia set**.

A **Mandelbrot set** is defined as

$$M = \{q \in \mathbf{H} \mid 0 \in K_p \text{ for } p(z) = z^2 + q\},$$

where the polynomial $p(z)$ is given by

$$p(z) = z^n + q,$$

and n is an integer.

Quaternion Julia sets extend beyond the complex plane. Consider a complex polynomial $p(z)$. It will have a Julia set J in the complex plane, and J will necessarily be contained in the quaternion Julia set of $p(z)$. But J could in fact be the entire Julia set in the quaternions as well. In other words, extending to the quaternions could provide us with nothing new. Fortunately, some Julia sets can clearly be seen to extend beyond the complex plane, forming truly four-dimensional objects. When K_p for $p(z) = z^2 + q$ is examined for q outside the Mandelbrot set M, it turns out that K_p is an infinite disjoint union of sets that we find difficult to fully describe. Thus we do not yet have a full analog to the classification theorem for Julia sets in terms of the Mandelbrot set, as in the complex case.

4.5 Differential Forms

Let F_1 be the set of $\mathbf{R}$-linear maps from $\mathbf{H}$ to $\mathbf{H}$.

$$F_1 = \{\alpha\colon \mathbf{H} \longrightarrow \mathbf{H} \mid \alpha \text{ is a linear map}\}.$$

We can define maps $\Gamma_r\colon F_1 \longrightarrow \mathbf{H}$ and $\Gamma_l\colon F_1 \longrightarrow \mathbf{H}$ as follows:

$$\Gamma_r(\alpha) = \alpha(1) + i\alpha(i) + j\alpha(j) + k\alpha(k)$$

and

$$\Gamma_l(\alpha) = \alpha(1) + \alpha(i)i + \alpha(j)j + \alpha(k)k.$$

A function $f\colon \mathbf{H} \longrightarrow \mathbf{H}$ is called **real-differentiable** if it is differentiable in the usual sense. Its differential at a point $q \in \mathbf{H}$ is an $\mathbf{R}$-linear map $df_q\colon \mathbf{H} \longrightarrow \mathbf{H}$. By identifying the tangent space at each point of $\mathbf{H}$ with $\mathbf{H}$ itself, we can regard the differential as a quaternion-valued 1-form

$$df = \frac{\partial f}{\partial t}dt + \frac{\partial f}{\partial x}dx + \frac{\partial f}{\partial y}dy + \frac{\partial f}{\partial z}dz. \tag{4.11}$$

The differential of the identity function $(f(q) = q = t + ix + jy + kz)$ is

$$dq = dt + idx + jdy + kdz, \tag{4.12}$$

which is regarded as an $\mathbf{R}$-linear transformation of $\mathbf{H}$. The exterior product of dq with itself is

$$dq \wedge dq = \varepsilon_{ijk} e_i dx_j \wedge dx_k = 2(idy \wedge dz + jdz \wedge dx + kdx \wedge dy), \tag{4.13}$$

where

$$\varepsilon_{ijk} = \begin{cases} 0 & \text{if any two indices are equal} \\ 1 & \text{for } (1,2,3) = (i,j,k) \\ \operatorname{sign}(\rho) & \text{for } (\rho_1, \rho_2, \rho_3) = (1,2,3). \end{cases}$$

In the above equation, $\rho = (\rho_1, \rho_2, \rho_3)$ is a permutation of $(1,2,3)$, and we denote the basic quaternions (i,j,k) by e_i, and the coordinates x, y, z by x_i, $i = 1,2,3$. For the constant real 4-form we use the abbreviation

$$v = dt \wedge dx \wedge dy \wedge dz. \tag{4.14}$$

An **oriented k-parallelepiped** in $\mathbf{H}$ is a map $C\colon I^k \longrightarrow \mathbf{H}$, where $I^k \subset R^k$ is the closed unit k-cube, of the form

$$C(t_1, \cdots, t_k) = q_0 + t_1 h_1 + \cdots + t_k h_k.$$

Here $q_0 \in \mathbf{H}$ is called the **original vertex** of the parallelepiped, and $h_1, \cdots, h_k \in \mathbf{H}$ are called its **edge-vectors**. An oriented k-parallelepiped is non-degenerate if its edge-vectors are linearly independent (over $\mathbf{R}$). A non-degenerate 4-parallelepiped is **positively oriented** if $v(h_1, \cdots, h_4) > 0$ and **negatively oriented** if $v(h_1, \cdots, h_4) < 0$, where v is the volume form defined in Equation 4.14.

The 3-form Dq (distinct from dq) is defined by the pairing

$$< h_1, D(h_2, h_3, h_4) > = v(h_1, h_2, h_3, h_4).$$

Thus, $Dq(i, j, k) = 1$, $Dq(1, e_i, e_j) = -\varepsilon_{ijk} e_k$ and the coordinate expression for Dq is

$$Dq = dx \wedge dy \wedge dz - i dt \wedge dy \wedge dz - j dt \wedge dz \wedge dx - k dt \wedge dx \wedge dy. \tag{4.15}$$

Geometrically, $Dq(a, b, c)$ is a quaternion which is perpendicular to a, b and c and has magnitude equal to the volume of the 3D parallelepiped whose edges are a, b and c.

Since the differential of a quaternion-valued function on $\mathbf{H}$ is an element of F_1, the map Γ_r can be applied to it:

$$\Gamma_r(df) = df(1) + i df(i) + j df(j) + k df(k).$$

The result is

$$\Gamma_r(df) = \frac{\partial f}{\partial t} + i\frac{\partial f}{\partial x} + j\frac{\partial f}{\partial y} + k\frac{\partial f}{\partial z}. \tag{4.16}$$

We introduce the following notation for the differential operator occurring in the previous equation, and for other related differential operators:

$$\begin{cases} \overline{\partial_l} f = \frac{1}{2}\Gamma_r(df) = \frac{1}{2}(\frac{\partial f}{\partial t} + e_i \frac{\partial f}{\partial x}) \\ \partial_l f = \frac{1}{2}(\frac{\partial f}{\partial t} - e_i \frac{\partial f}{\partial x}) \\ \overline{\partial_r} f = \frac{1}{2}\Gamma_r(df) = \frac{1}{2}(\frac{\partial f}{\partial t} + \frac{\partial f}{\partial x} e_i) \\ \partial_r f = \frac{1}{2}(\frac{\partial f}{\partial t} - \frac{\partial f}{\partial x} e_i) \\ \triangle f = \frac{\partial^2 f}{\partial t^2} + \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2}. \end{cases} \tag{4.17}$$

Note that ∂_l, $\overline{\partial_l}$,∂_r, and $\overline{\partial_r}$ are all commutative, and that

$$\triangle = 4\partial_r\overline{\partial_r} = 4\partial_l\overline{\partial_l}. \tag{4.18}$$

4.6 Regular Functions

Definition 4 *A function* $f\colon \mathbf{H} \longrightarrow \mathbf{H}$ *is* **quaternion-differentiable on the left** *at* q *if the limit*

$$\frac{df}{dq} = \lim_{h \to 0} \left(\frac{1}{h}(f(q+h) - f(q)) \right)$$

exists. A function $f\colon \mathbf{H} \longrightarrow \mathbf{H}$ is **quaternion-differentiable on the right** *at q if the limit*

$$\frac{df}{dq} = \lim_{h\to 0}\left((f(q+h) - f(q))\frac{1}{h}\right)$$

exists.

The above definition of the quaternionic derivative is too strong to have interesting consequences. In fact, one can prove the following result.

Theorem 17 *Suppose the function f is defined and quaternion-differentiable on the left throughout a connected open set U. Then on U, f has the form*

$$f(q) = a + bq$$

for some $a, b \in \mathbf{H}$.

See [37] for a detailed proof.

We now introduce the definition of regular quaternionic functions, which leads to a development similar to the theory of regular functions of a complex variable.

Definition 5 *A function $f\colon \mathbf{H} \longrightarrow \mathbf{H}$ is* **left-regular** *at $q \in \mathbf{H}$ if it is real-differentiable at q and there exists a quaternion $f_l'(q)$ such that*

$$d(dq \wedge dqf) = Dqf_l'(q). \tag{4.19}$$

It is **right-regular** *if there exists a quaternion $f_r'(q)$ such that*

$$d(fdq \wedge dq) = f_r'(q)Dq. \tag{4.20}$$

Clearly, the theory of left-regular functions will be equivalent to the theory of right-regular functions. For simplicity, we will only consider left-regular functions, which we will call simply **regular**. We write $f_l'(q) = f'(q)$ and call it the **derivative** of f at q.

An application of Stokes's theorem gives the following characterization of the derivative of a regular function as the limit of a difference quotient:

Proposition 18 *Suppose that f is regular at q_0 and continuously differentiable in a neighborhood of q_0. Then given $\varepsilon > 0$, there exists $\delta > 0$ such that if C is a non-degenerate oriented 3-parallelepiped with $q_0 \in C(I^3)$ and $q \in C(I^3) \Rightarrow |q - q_0| < \delta$, then*

$$\left|\left(\frac{1}{\int_c Dq}\right)\left(\int_{\partial C} dq \wedge dqf\right) - f'(q_0)\right| < \varepsilon.$$

The following proposition is another property of regular functions.

Proposition 19 (the Cauchy-Riemann-Fueter equations) *A real-differentiable function f is regular at q if and only if $\overline{\partial}_l f = 0$. That is,*

$$\frac{\partial f}{\partial t} + i\frac{\partial f}{\partial x} + j\frac{\partial f}{\partial y} + k\frac{\partial f}{\partial z} = 0. \tag{4.21}$$

From Theorem 17, it follows that if f is regular and twice differentiable, then

$$\triangle f = 0, \tag{4.22}$$

that is, f is harmonic. We will see that a regular function is necessarily infinitely differentiable, so all regular functions are harmonic.

4.7 Cauchy's Theorem and the Integral Formula

The integral theorems for regular quaternionic functions have as wide a range of validity as those for regular complex functions – considerably wider than integral theorems for harmonic functions. Cauchy's theorem holds for any rectifiable contour of integration; the integral formula is similar to Poisson's formula in that both give the values of a function in the interior of a region in terms of its values on the boundary. The integral formula holds for a general rectifiable boundary and thus constitutes an explicit solution to the general Dirichlet problem.

The algebraic basis of these theorems is the equation

$$\begin{aligned} d(gDqf) &= dg \wedge Dqf - gDq \wedge df \\ &= \{(\overline{\partial}_r g)f + g(\overline{\partial}_l f)\}v. \end{aligned} \tag{4.23}$$

Note,

$$dg \wedge Dq = \overline{\partial}_r g$$

and

$$Dq \wedge Df = \overline{\partial}_l f.$$

Proposition 20 *A differentiable function f is regular at q if and only if*

$$Dq \wedge df_q = 0.$$

From the above proposition, together with Stokes's theorem, it follows that if f is regular and continuously differentiable in a domain D with a differentiable boundary, then

$$\int_{\partial D} Dqf = 0.$$

As in complex analysis, however, the conditions on f can be weakened by using Goursat's dissection argument. Applying this to a parallelepiped, we obtain the following lemma.

Lemma 21 *If f is regular at every point of the 4-parallelepiped C,*

$$\int_{\partial C} Dqf = 0. \tag{4.24}$$

The dissection argument can also be used to prove the Cauchy-Fueter integral formula for a parallelepiped:

Lemma 22 *If f is regular at every point of the positively oriented 4-parallelepiped C, and q_0 is a point in the interior of C,*

$$f(q_0) = \frac{1}{2\pi^2} \int_{\partial C} \frac{(q - q_0)^{-1}}{|q - q_0|^2} Dqf(q). \tag{4.25}$$

The following theorem makes it valid to apply Stokes's theorem and so obtain Cauchy's theorem for the boundary of any differentiable 4-chain.

Theorem 23 *A function which is regular in an open set U is real-analytic in U.*

We also need some definitions and notation.

Definition 6 *Let $C\colon I^3 \longrightarrow \mathbf{H}$ be a continuous map of the unit 3-cube into $\mathbf{H}$, and let $P : 0 = s_0 < s_1 < \cdots < s_p = 1$, $Q : 0 = t_0 < t_1 < \cdots < t_q = 1$ and $R : 0 = u_0 < u_1 < \cdots < u_r = 1$ be three partitions of the unit interval I. Define*

$$\sigma(C; P, Q, R) = \sum_{l=0}^{p-1} \sum_{m=0}^{q-1} \sum_{n=0}^{r-1} DqV_{l,m,n}$$

where

$$V_{l,m,n} = (C(s_{l+1}, t_m, u_n) - C(s_l, t_m, u_n), C(s_l, t_{m+1}, u_n) - C(s_l, t_m, u_n), C(s_l, t_m, u_{n+1}) - C(s_l, t_m, u_n)).$$

C is a rectifiable 3-cell. Suppose there is a real number M such that $\sigma(C; P, Q, R) < M$ for all partitions P, Q, R. If this is the case, then the least upper bound of the numbers $\sigma(C; P, Q, R)$ is called the **content** *of C and is denoted by $\sigma(C)$.*

Definition 7 *Let f and g be quaternion-valued functions defined on $C(I^3)$. Let*

$$S = \sum_{l=0}^{p-1} \sum_{m=0}^{q-1} \sum_{n=0}^{r-1} f(C(\overline{s}_l, \overline{t}_m, \overline{u}_n) DqV_{l,m,n} g(C(\overline{s}_l, \overline{t}_m, \overline{u}_n),$$

where

$$V_{l,m,n} = (C(s_{l+1}, t_m, u_n) - C(s_l, t_m, u_n), C(s_l, t_{m+1}, u_n) - C(s_l, t_m, u_n), C(s_l, t_m, u_{n+1}) - C(s_l, t_m, u_n))$$

and $s_l \leq \overline{s}_l \leq s_{l+1}$, $t_m \leq \overline{t}_m \leq t_{m+1}$, and $u_n \leq \overline{u}_n \leq u_{n+1}$. We say that $fDqg$ is integrable over C if the sum S has a limit in the sense of Riemann-Stieltjes integration as $|P|, |Q|, |R| \to 0$, where

$$|P| = \max_{0 \leq l \leq p-1} |s_{l+1} - s_l|$$

measures the coarseness of the partition P. If this limit exists, we denote it by $\int_C fDqg$.

We extend these definitions to define rectifiable 3-chains and integrals over rectifiable 3-chains in the usual way.

Definition 8 *Let C be a 3-chain in $\mathbf{H}$. C is a rectifiable 3-chain if there is a dissection of C such that, $C = \sum_n C_n$ and C_n are 3-cells. We denote the content of C by $\sigma(C) = \sum_n \sigma(C_n)$.*

Definition 9 *Let f and g be quaternion-valued functions defined on a 3-chain C. We say that $fDqg$ is integrable over C if $fDqg$ is integrable over C_n, where $C = \sum_n C_n$ is a dissection of C, and the C_n are 3-cells. We denote $\sum_n \int_{C_i} fDqg$ by $\int_C fDqg$.*

Just as for rectifiable curves, we can show that $fDqg$ is integrable over 3-chains C if f and g are continuous and C is rectifiable, and

$$\left|\int_C fDqg\right| \leq \left(\max_C |f|\right)\left(\max_C |g|\right)\sigma(C).$$

Furthermore, we have the following weak form of Stokes's theorem:

Theorem 24 (Stokes's Theorem For a Rectifiable Contour) *Let C be a rectifiable 3-chain in $\mathbf{H}$ with $\partial C = 0$, and suppose f and g are continuous functions defined in a neighborhood U of the image of C, and that $fDqg = d\omega$ where ω is a 2-form on U. Then*

$$\int_C fDqg = 0.$$

We can now give the most general forms of Cauchy's theorem and the integral formula.

Theorem 25 *Suppose f is regular in an open set U, and let C be a rectifiable 3-chain which is homologous to 0 in the singular homology of U. Then*

$$\int_C Dqf = 0.$$

For a general form of the integral formula, we need an analogue of the notion of the winding number of a curve around a point in the plane. Let q be any quaternion, and let C be a 3-cycle in $\mathbf{H} - \{q\}$. Then C is homologous to $n\partial C_0$, where C_0 is a positively oriented 4-parallelepiped in $\mathbf{H} - \{q\}$, and n is an integer (independent of the choice of C_0), which we will call the *wrapping number of C about q*.

Theorem 26 (The integral formula for a rectifiable contour) *Suppose f is regular in an open set U. Let q_0 be a point in U, and let C be a rectifiable 3-chain which is homologous, in the singular homology of $U - \{q_0\}$, to a differentiable 3-chain whose image is ∂B for some ball $B \in U$, then*

$$\frac{1}{2\pi^2}\int_C \frac{(q-q_0)^{-1}}{|q-q_0|^2} Dqf(q) = nf(q_0),$$

where n is the wrapping number of C about q_0.

Many of the standard theorems of complex analysis depend only on Cauchy's integral formula, and so they also hold for quaternionic regular functions. An obvious example is the Maximum-Modulus Theorem.

Theorem 27 (Maximum-Modulus Theorem) *Suppose f is regular on an open region D in $\mathbf{H}$ and f is not constant. Let $D' \subset D$ be an open region, $\partial D' = C$. If $|f'(q)| < M$ on C, then $|f(q)| < M$ at all points in D'.*

4.8 Linear and Quadratic Regular Functions

The properties of the regular functions are certainly desirable, but it is somewhat disappointing that this class does not even contain the identity function $f(z) = z$, or any other polynomial in z. What do the regular functions look like?

First, let us see why a left quaternion linear function $f(z) = az$ is not a regular function, where a is a constant quaternion. Note $\partial f(z) = -2a$ is zero if and only if $a = 0$. Thus, if we wish to consider non-trivial regular linear functions, we must drop the quaternion linearity condition and instead impose the requirement of real-homogeneity.

The most general left-regular function that is real-homogeneous of degree 1 is

$$f(z) = (iz + zi)a + (jz + zj)b + (kz + zk)c, \tag{4.26}$$

where a, b, c are constant quaternions. Is $f^n(z)$ regular for all $n = 1, 2, 3, ...$? By Briggs [7], a sufficient condition for $f^n(z)$ to be regular is that b and c are both real multiples of a. Then Equation 4.26 can be represented as a map on $\mathbf{R}^4$. In an appropriate basis, the map is equivalent to a map in the complex plane. As such, it loses much of its interest from a quaternion point of view.

Second, let us see whether anything more interesting is possible with quadratic quaternion functions. Sudbery [37] shows that a basis for such functions is $\{P_{11}, P_{22}, P_{33}, P_{12}, P_{13}, P_{23}\}$, where

$$\begin{aligned} P_{11}(z) &= (q_0 i - q_1)^2 \\ P_{22}(z) &= (q_0 j - q_2)^2 \\ P_{33}(z) &= (q_0 j - q_3)^2 \\ P_{12}(z) &= q_1 q_2 - q_0(iq_2 + jq_1) \\ P_{12}(z) &= q_1 q_3 - q_0(iq_3 + kq_1) \\ P_{12}(z) &= q_2 q_3 - q_0(iq_3 + kq_2). \end{aligned} \tag{4.27}$$

Thus, the most general left-regular map that is real-homogeneous of degree 2 is

$$f(z) = P_{11}(z)a + P_{22}(z)b + P_{33}(z)c + P_{12}(z)d + P_{13}(z)e + P_{23}(z)f, \tag{4.28}$$

where a, b, c, d, e, f are constant quaternions. By [7], the dynamics of most of these functions can be understood with the known theory of complex analytic dynamics. So the requirement of regular iterability (all iterates of a function are regular) is too strong to allow the existence of functions other than those equivalent to complex quadratic functions.

4.9 Difficulties of the Quaternion Analytic Proof of Distance Estimation

An initial attempt to prove distance estimation for quaternion Julia sets and Mandelbrot sets might use regular functions and the theorems stated in the previous sections. However, as shown in the last section, the set of regular functions is much smaller than the set of analytic functions in complex space. Many good functions, for example the polynomial $f(z) = z^2$, are not regular. And the functions we might need in the proof may not be the regular functions at all. All these facts make the analogous proof very difficult, if not impossible. Let us look at the idea in detail.

As we know, even though the Möbius transformation defined by

$$M(z) = (az + b)(cz + d)^{-1} \tag{4.29}$$

preserves regularity by [37] under certain conditions, a composite function or a product function of two regular quaternion functions may not be a regular function. This means the quaternionic analogues to Schwarz's Lemma and Koebe's $\frac{1}{4}$ Theorem may not hold for quaternion regular functions. But these were the keys to the proof of the distance estimation formula. And the regularity of a composite or a product function of two regular functions was used in many places in the proofs of those theorems.

For example, to prove Schwarz's Lemma, we might need the function $g(z) = z^{-1}f(z)$ to be a regular function. In the complex case, if $f(z)$ is analytic on a disk D, then $g(z) = z^{-1}f(z)$ is analytic on $D - \{0\}$. However, in the quaternion case $g(z)$ may not be a regular function at all, even if $f(z)$ is a regular function.

Based on the above discussion, an analogous proof of distance estimation by quaternion analysis is quite difficult. For this reason, we set aside quaternion analysis and take another look at the problem from the perspective of high-dimensional geometry. This geometric approach is the subject of Chapter 7.

Chapter 5

Quaternions and the Dirac String Trick

The purpose of this small chapter is to form an interlude between the complexities of quaternionic analysis and the geometry of iteration in the quaternions. Sir William Rowan Hamilton discovered/invented the quaternions with the intent of finding an algebra related to three dimensional space that would be as informative as the complex numbers had been for two dimensional space. It turns out that the quaternions are a good algebra for understanding rotations in three and four dimensional space. They also have a remarkable topological interpretation that is related to three dimensional space and to the properties of a twisted band in 3-space. This topological interpretation is also related to certain aspects of quantum physics. We give a short introduction to these ideas here. For more a more detailed discussion see [21] Part 2, Section 10.

Recall that the quaternions are an algebraic system with generating elements i, j, k such that

$$i^2 = j^2 = k^2 = ijk = -1$$

and that it follows from these equations and the associativity of the multiplication of the quaternions that

$$ij = k,\ jk = i,\ ki = j$$

while

$$ji = -k,\ kj = -i\ and\ ik = -j.$$

Not only are i, j and k square roots of minus one, but in fact so is any linear combination $ai + bj + ck$ where $a^2 + b^2 + c^2 = 1$ and a, b, c are real numbers. Because of this, it is convenient to think of the quaternions i, j and k as the three basic directions in three dimensional space and the scalars of the form d (d a real number) as directions in a fourth dimension. Thus a general quaternion is of the form $ai+bj+ck+d$ for real numbers a, b, c, d and describes a point in four dimensional space (a, b, c, d). Any unit direction $ai + bj + ck$ in 3-space yields a square root of minus one.

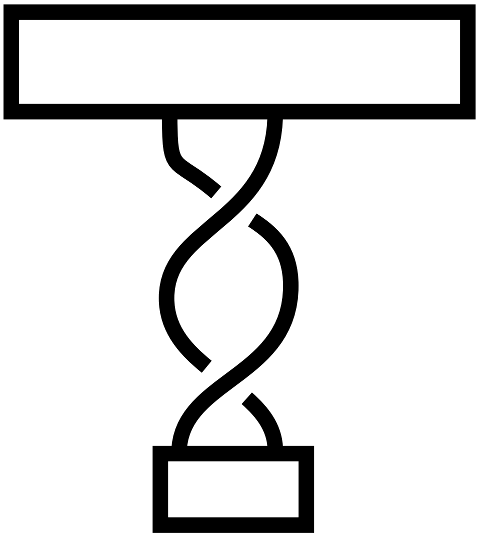

Figure 5.1: Twisted Ribbon

It is the purpose of this section to point out that there is a topological interpretation of this phenomenon of square roots of minus one in every spatial direction, and to show that the rules for the quaternion multiplications $i^2 = j^2 = k^2 = ijk = -1$ are a description of the way ribbons can twist in three dimensional space! Thus the basics of quaternion multiplication actually have a clear three dimensional interpretation if we allow a little topology to enter into the discussion.

Let's begin with a ribbon attached to a wall and to a ball hanging in midair as shown in Figure 5.1, where the wall is represented by a rectangle at the top of the figure, and the ball is represented by a rectangle at the bottom of the figure.

By turning the ball we can put a twist in the ribbon. In Figure 5.1 we show a twist by 360 degrees. We shall call a 180 degree twist a *half twist* of the ribbon and the 360 degree twist a (right handed) *full twist* of the ribbon. Now view Figure 5.2. Here we show the result of moving the ribbon around the ball, keeping its endpoints fixed to both the ball and the wall. The ball does not rotate in this deformation. We start with a right-handed full twist in the ribbon. After the deformation, we have a full *left handed* twist in the ribbon. Moving the ribbon around the ball converts a full right handed twist to a full left handed twist.

Now, we draw the consequence of this conversion of right handed twist to left handed twist. We start with a 720 degree twist in the ribbon and regard it as two full 360 degree twists. We then move the ribbon around the ball, converting one of the right full twists to a left full twist. The band now has one right full twist followed by a left full twist upon it, and these cancel one another. As a result, all the twist on the band is gone! A 720 degree twist in the band is reduced to no twist at all by a movement of the band around the ball. This is the *belt trick* or *Dirac string trick.*

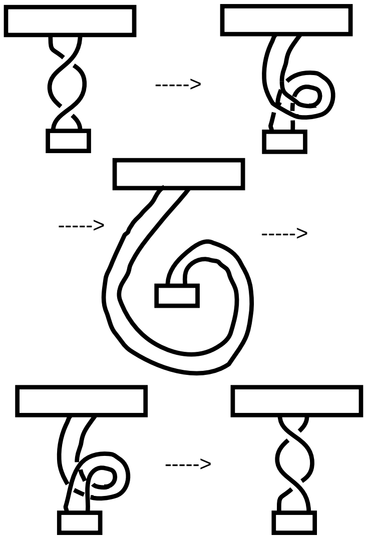

Figure 5.2: Converting a Right Twist to a Left Twist

The belt trick tells us that the operation of a single (180 degree) twist has order four from the point of view of the topology of a band attached to a wall and a ball. Four 180 degree turns result in no turn at all. In this way we can interpret a single 180 degree turn as an instance of $i = \sqrt{-1}$ with i^2 representing a 360 twist (either left or right since they are topologically equivalent). Then we have i^3 is a non-trivial twist, but $i^4 = 1$ in the sense that its band has no twist at all. Via this train of thought, we identify the 360 twist on the band as "-1" since the square of this twist is 720 degrees, which is equivalent to no twist at all.

Now think of the band and its ball as representing a spatial direction. We twist around this given spatial direction, and we find that there is a square root of minus one hidden in the topology of a 180 degree twist around that axis. This phenomenon occurs for every spatial direction. There was nothing special about the particular direction that we chose. So we have arrived at a topological interpretation of the quaternionic phenomenon of one square root of minus one for every direction in three dimensional space.

The belt trick provides a topological/mechanical construction for the operations among the quaternions i, j and k. Attach a belt to a fixed wall and replace the ball by a movable card. Let i, j and k denote turns of the card by 180 degrees about perpendicular axes in three space. After such turns, let the state of the belt be reduced (if this is possible) by the use of the belt trick with the ends of the belt fixed to the wall and to the card. The belt may be moved around the card without twisting the card. There are other hand movements that are related to the cancellation of 720 degrees of twist. One well-known version is the so-called plate trick where one can hold a plate flat on the palm of the hand and turn the hand by 360 degrees while moving it downward and by another 360 degrees in the same turning direction while moving upward. In the end, one's arm is in exactly the same position as before, even though the hand has turned fully through 720 degrees.

The belt trick and its relation to the quaternions is part of a deeper connection between the quaternions and quantum physics. It turns out that an observer and an electron are linked physically in a way that is similar in structure to the ball and the wall connected by a ribbon. If an observer were to move around an electron by 360 degrees, she would return to the same place in space, but her physical state relative to the electron would not be the same. Two full turns of the observer however, do return her to the original physical state. In the mathematics of quantum mechanics, this phenomenon is modelled in terms of properties of the symmetry group of the system, which is the group $SU(2)$ rather than the group of rotations of three space. The group $SU(2)$, so important for quantum physics, is isomorphic with the quaternions of unit length in four dimensional space!

Part III

Hypercomplex Iterations

Chapter 6

Quaternion Mandelbrot Sets

6.1 Quaternion Mandelbrot Sets

Recall that the Mandelbrot set is defined as follows: To each point q in the quaternions, we associate the function $f_q(z) = z^2 + q$ and consider the iteration of $f_q(z)$ applied to zero. If this iteration does not go to infinity, then q is a member of the Mandelbrot set:

$$M = \{q \in \mathbf{H} \mid f_q^n(0) \text{ remains bounded as } n \to \infty\}.$$

The lack of commutativity of the quaternions makes a proof of distance estimation using quaternion analysis more difficult than in the complex case. However, the distance estimation formula for the Mandelbrot set can be proved more easily than the analogous formula for Julia sets. In the next section, a proof of distance estimation for quaternion Mandelbrot sets is given. Here, we use $f(z) = z^2 + q$ as the iterated function. A parallel discussion and proof can be obtained directly for $f(z) = z^n + q$.

6.2 The Distance Estimate for Quaternion Mandelbrot Sets

Let M_4 denote the quaternion Mandelbrot set generated by the function $f(z) = z^2 + q$ in $\mathbf{H}$, and M_2 is the complex Mandelbrot set in $\mathbf{C}$. Let $q \in \mathbf{H}$. According to Hamilton [14] (or, see Section 4.2), there is a quaternion value $p \in \mathbf{H}$, $|p| = 1$ such that $p^{-1}qp = c$ and $c \in \mathbf{C}$. Let us call c the **corresponding complex value** for q.

Note that capital Z will denote a quaternion variable, while lower case z will denote a complex variable in this chapter.

Lemma 28 *For any quaternion value $q \in \mathbf{H}$ and its corresponding complex value $c \in \mathbf{C}$, the quaternion $q \in M_4$ if and only if $c \in M_2$.*

Proof: Let $p \in \mathbf{H}$, $|p| = 1$ such that $p^{-1}qp = c$. Let $z_0 = Z_0 = 0$ be the initial value for

$$f_c(z) = z^2 + c \tag{6.1}$$

and

$$f_q(Z) = Z^2 + q. \tag{6.2}$$

Let $z_1, z_2, \ldots, z_n$ and $Z_1, Z_2, \ldots, Z_n$ be the orbits of z_0 and Z_0 under f_c and f_q, respectively. So

$$z_1 = c$$

and

$$Z_1 = q = pcp^{-1} = pz_1p^{-1};$$
$$z_2 = c^2 + c = f_c(z_1)$$

and

$$Z_2 = q^2 + q = p(c^2 + c)p^{-1} = pz_2p^{-1};$$
$$\vdots$$
$$z_n = f_c(z_{n-1})$$
$$Z_n = pz_np^{-1}.$$

Therefore,

$$|Z_n| = |z_n|.$$

This means $q \in M_4$ if and only if $c \in M_2$. □

Lemma 29 *For any quaternion value $q \in \mathbf{H}$ and any quaternion value $p \in \mathbf{H}$ with $|p| = 1$, the quaternion $q \in M_4$ if and only if $pqp^{-1} \in M_4$.*

Proof: Let $z_0 = Z_0 = 0$ be the initial value for

$$f_q(z) = z^2 + q \tag{6.3}$$

and

$$f_{pqp^{-1}}(Z) = Z^2 + pqp^{-1}. \tag{6.4}$$

Let $z_1, z_2, \ldots, z_n$ and $Z_1, Z_2, \ldots, Z_n$ be the orbits of z_0 and Z_0 under f_q and $f_{pcp^{-1}}$ respectively. So,

$$z_1 = q$$

and

$$Z_1 = pqp^{-1} = pz_1p^{-1};$$
$$z_2 = q^2 + q$$

and

$$Z_2 = (pqp^{-1})^2 + pqp^{-1} = p(q^2 + q)p^{-1} = pz_2p^{-1};$$
$$\vdots$$
$$Z_n = pz_np^{-1}.$$

Therefore,

$$|Z_n| = |z_n|.$$

This means $q \in M_4$ if and only if $pqp^{-1} \in M_4$. □

Lemma 30 *For any quaternion value $p \in \mathbf{H}$ with $|p| = 1$,*

$$Re(pip^{-1}) = 0. \tag{6.5}$$

Proof: Let

$$p = a + bi + cj + dk.$$

Since $|p| = 1$,

$$\begin{aligned} p^{-1} &= a - bi - cj - dk \\ pip^{-1} &= (a + bi + cj + dk)i(a - bi - cj - dk) \\ &= (ai - b - ck + dj)(a - bi - cj - dk). \end{aligned}$$

So,

$$\mathrm{Re}(pip^{-1}) = -ab - abi^2 + cdk^2 - cdj^2 = 0$$

and

$$\mathrm{Pu}(pip^{-1}) = (a^2 + b^2 - c^2 - d^2)i + 2(bc + ad)j + 2(bd - ac)k.$$

□

Lemma 31 *For any $c \in \mathbf{C}$, the distance from c to M_4 is equal to the distance from c to M_2, that is*

$$d(c, M_4) = d(c, M_2). \tag{6.6}$$

Proof: Suppose

$$d(c, M_2) > d(c, M_4). \tag{6.7}$$

Let $c = a + bi$. Let $c_0 \in \mathbf{C}$ be a point in M_2 such that $d(c, c_0) = d(c, M_2)$. Let $q_0 \in \mathbf{H}$ be a point in M_4 with $q_0 \notin \mathbf{C}$ such that $d(c, q_0) = d(c, M_4)$. Set $q_0 = x_0 + y_0 u$, where $u = u_1 i + u_2 j + u_3 k$, $u_1^2 + u_2^2 + u_3^2 = 1$. We can assume that $|u_1| < 1$, namely one of u_2 and u_3 is not 0. Then

$$\begin{aligned} d^2(c, q_0) &= (a - x_0)^2 + (b - u_1 y_0)^2 + (u_2 y_0)^2 + (u_3 y_0)^2 \\ &= (a - x_0)^2 + b^2 + y_0^2 - 2bu_1 y_0. \end{aligned}$$

Let $c_0' \in \mathbf{C}$ be the corresponding complex value of q_0. That is, there is a quaternion $p \in \mathbf{H}$, $|p| = 1$ such that $c_0' = p^{-1} q_0 p$. By Lemma 28, $c_0' \in M_2$. Set $c_0' = x + yi$, then

$$x_0 + y_0 u = q_0 = pc_0' p^{-1} = p(x + yi)p^{-1} = x + ypip^{-1},$$

so by Lemma 6.2, $x_0 = x$, and $y_0 = y$. Therefore,

$$d^2(c, c_0') = (a - x_0)^2 + (b - y_0)^2 = (a - x_0)^2 + b^2 + y_0^2 - 2by_0.$$

So by the fact that $|u_1| < 1$

$$d(c, c_0) > d(c, q_0) > d(c, c_0').$$

This contradicts our assumption that $d(c, c_0) = d(c, M_2)$. □

Lemma 32 *For any quaternion value $q \in \mathbf{H}$ and its corresponding complex value $c \in \mathbf{C}$,*

$$d(q, M_4) = d(c, M_2). \tag{6.8}$$

Proof: Suppose

$$d(q, M_4) \neq d(c, M_2). \tag{6.9}$$

Let $p \in \mathbf{H}$, $|p| = 1$, be the quaternion such that $c = p^{-1}qp$. Let $q_0 \in M_4$ such that $d(q, q_0) = d(q, M_4)$, but $p^{-1}q_0p \notin \mathbf{C}$. Let $c_0 \in M_2$ such that $d(c, c_0) = d(c, M_2) = d(c, M_4)$.

Since $q_0 \in M_4$ and $c_0 \in M_2$, by Lemma 6.2, we have $p^{-1}q_0p \in M_4$ and $pc_0p^{-1} \in M_4$.

So,

$$d(q, q_0) = |q - q_0| = |p^{-1}qp - p^{-1}q_0p| = |c - p^{-1}q_0p| > d(c, M_4) = |c - c_0|,$$

but

$$|c - c_0| = |pcp^{-1} - pc_0p^{-1}| = |q - pc_0p^{-1}| = d(q, pc_0p^{-1}).$$

Therefore,

$$d(q, q_0) > d(q, pc_0p^{-1}).$$

This contradicts the assumption that $d(q, q_0) = d(c, M_4)$. □

Theorem 33 *The distance from a point q outside the quaternion Mandelbrot set M_4 to M_4 is bounded by*

$$d_- = \frac{|Z_n|}{2|Z_n|^{\frac{1}{2^n}}|Z'_n|} \ln |Z_n| < d(q, M_4) < d_+ = \frac{|Z_n|}{|Z'_n|} \ln |Z_n|, \tag{6.10}$$

where $Z_n = F^n(Z_0)$, $Z_0 = 0$ and F is defined as $F(Z) = Z^2 + q$, where $Z, q \in \mathbf{H}$.

Proof: Let $f(z) = z^2 + c$, where c is the corresponding complex value of q, and let $z_0 = Z_0 = 0$ be the initial value for $f(z)$ and for $F(Z)$. By the proof of Lemma 6.2,

$$|z_n| = |Z_n|, \tag{6.11}$$

where $n = 0, 1, 2, \ldots$ So, by Theorem 2,

$$\frac{|z_n|}{2|z_n|^{\frac{1}{2^n}}|z'_n|} \ln |z_n| < d(c, M_2) = d(q, M_4) < \frac{|z_n|}{|z'_n|} \ln |z_n|.$$

Therefore,

$$d_- = \frac{|Z_n|}{2|Z_n|^{\frac{1}{2^n}}|Z'_n|} \ln |Z_n| < d(q, M_4) < d_+ = \frac{|Z_n|}{|Z'_n|} \ln |Z_n|.$$

□

Chapter 7

Distance Estimation in Higher Dimensional Spaces

7.1 Higher Dimensional Deterministic Fractals

In this section we point out a natural generalization of the quaternionic iteration that leads to an interesting class of higher dimensional fractals. The generalization also provides a way to look at the structure of the three and four dimensional quaternion fractals. (In this section lower case z will refer to a point in $\mathbf{R}^{N+1}$, as explained below.)

Let

$$\mathbf{R}^{N+1} = \{a + bu \mid a, b \in \mathbf{R} \text{ and } u \in S^{N-1}\}$$

where $S^{N-1} = \{u \in \mathbf{R}^N \mid u_1^2 + u_2^2 + ... + u_N^2 = 1\}$. That is, we take

$$\mathbf{R}^N = \{x \mid x = (x_1, x_2, ..., x_N),\ x_i \in \mathbf{R}\},$$

and we write $a + bu = (a, bu_1, bu_2, ..., bu_N)$ for a point in $\mathbf{R}^{N+1}$.

Now define $u^2 = -1$ for each $u \in S^{N-1}$. This gives us a rule for taking powers $(a+bu)^k$ via the formula

$$(a + bu)(c + du) = (ac - bd) + (bc + ad)u.$$

Thus,

$$(a + bu)^2 = (a^2 - b^2) + (2ab)u.$$

Note that in this generalization we do not have a formula for multiplying $(a+bu)(c+dv)$ when $u \neq v$, though such formulas exist in the complex numbers and quaternions. For example in the quaternions, u and v are vectors in the two-dimensional sphere S^2 in three-dimensional space and, as vectors in three-dimensional space,

$$uv = -u \cdot v + u \times v.$$

Note that in this well-known formula for the multiplication of two 3-space quaternions, the component in the fourth direction is indicated by the addition of a scalar $(-u \cdot v)$ to a vector

$(u \times v)$. Thus our general formula for points in $\mathbf{R}^{N+1}$ and their powers is a generalization of the quaternion formalism.

With this picture of $\mathbf{R}^{N+1}$, define

$$F_k : \mathbf{R}^{N+1} \to \mathbf{R}^{N+1}$$

by the formula

$$F_k(a + bu) = (a + bu)^k + (c + dv)$$

where $c + dv = q$ is a chosen vector. (Here $v \neq u$ is possible.) Hence,

$$F_k(z) = z^k + q. \tag{7.1}$$

This function is the analog for $\mathbf{R}^{N+1}$ of the corresponding function in the quaternions and in the complex numbers.

The corresponding iterations, Julia sets, and Mandelbrot sets can be defined, computed and studied just as in the complex and quaternionic cases. In particular, we can write k-th root functions just as in the complex numbers and use inverse iteration to produce point cloud images of the sets. The generalized distance estimation techniques of the next section apply, and they will allow us to compute and render the images.

It is worth noting that in the case of $\mathbf{R}^3 = \{a + bu \mid u \in S^1\}$, this formulation gives a direct three-dimensional picture of the restricted quaternion mappings that produce our 3D quaternionic fractal images.

Note that the quaternions are represented as $\mathbf{R}^4 = \{a + bu \mid a, b \in \mathbf{R},\, u \in S^2\}$, where $u = \alpha i + \beta j + \gamma k$ with $\alpha, \beta, \gamma \in \mathbf{R}$ and $\alpha^2 + \beta^2 + \gamma^2 = 1$. In this case, if $u, v \in S^2$, we can define $uv = -u \cdot v + u \times v$ where $u \cdot v$ is the standard dot product in $\mathbf{R}^3$ and $u \times v$ is the vector cross product. This gives quaternion multiplication. Note that $u^2 = -1$ for all $u \in S^2$.

7.2 The Cayley Numbers

The Cayley numbers are a generalization of the quaternions to an algebraic structure on eight dimensional space. Just as complex numbers can be regarded as pairs of real numbers and quaternions can be regarded as pairs of complex numbers, Cayley numbers can be regarded as pairs of quaternions. We write a Cayley number in the form $a + Jb$ where a and b are quaternions and J is a new square root of minus one. (The basic generators of the quaternions are i, j and k.) Recall that we have the operation $z \longrightarrow \overline{z}$ that *conjugates* a quaternion. The conjugate of $t + ip + jq + kr$ is $t - ip - jq - kr$. With this operation in mind, we can define Cayley multiplication by the following formula:

$$(a + Jb)(c + Jd) = (ac - d\overline{b}) + J(cb + \overline{a}d).$$

This definition is constructed very carefully so that we have

$$(a + Jb)(\overline{a} - Jb) = (a\overline{a} + b\overline{b}).$$

This means that non-zero Cayley numbers have inverses. However, the Cayley numbers are not associative. For example $(Jb)c = J(cb)$ is not equal to $J(bc)$. It turns out that there is no way to extend the quaternions to an eight dimensional algebra that has inverses without encountering non-associativity, and matters degenerate even further after that.

Nevertheless, the Cayley numbers are a direct analogue of the quaternions and they still have the property that every Cayley number can be written in the form $a + bu$, where a and b are real numbers and u is a unit Cayley number in 7-space such that $u^2 = -1$. Thus the Cayley numbers provide a useful ground for generating higher dimensional hypercomplex fractals.

7.3 Distance Estimation in Higher Dimensional Spaces

The lack of commutativity of the quaternions seems to make the proof of quaternionic or higher dimensional distance estimation much more difficult than in the complex case. We would have much difficulty trying to mimic the method that uses the Riemann Mapping Theorem and other theorems of complex analysis. We now present our new approach to distance estimation.

Assume we are given a function

$$F : \mathbf{R}^N \longrightarrow \mathbf{R}^N.$$

Let J denote the Julia set of F. That is J is the frontier of the set of points that escape to infinity under the iteration of F. We shall say that a point z' is outside J (not to be confused with the set-theoretic complement of J) if $F^n(z) \to \infty$ as $n \to \infty$. We shall assume that J is a compact subset of $\mathbf{R}^N$ (and hence closed and bounded).

Given a point z outside J, we wish to estimate the distance δ from z to the set J. Since $\delta = Dist(z, J)$, there is a point $z_0 \in J$ and a unit vector $u \in \mathbf{R}^N$ such that $z_0 = z + \delta u$. We know that for n sufficiently large, $|F^n(z)|$ will be arbitrarily large. On the other hand, $|F^n(z_0)|$ is bounded by the maximal distance of points in the compact set J to the origin in $\mathbf{R}^N$. Therefore, for sufficiently large n, we have

$$\frac{|F^n(z)|}{|F^n(z_0) - F^n(z)|} \approx 1.$$

In fact, we can write the equation

$$\lim_{n\to\infty} \frac{|F^n(z)|}{|F^n(z_0) - F^n(z)|} = 1.$$

The degree of the approximation to 1 can be made arbitrarily small for sufficiently large n. See Figure 1.3 in Chapter 1.

Therefore,

$$\lim_{n\to\infty} \frac{|F^n(z)|}{|\frac{F^n(z+\delta u)-F^n(z)}{\delta}|} = \delta,$$

for $\delta > 0$. Let

$$D_{\delta,u}(G) = \frac{G(z+\delta u) - G(z)}{\delta}.$$

For any $G : \mathbf{R}^N \longrightarrow \mathbf{R}^N$, $D_{\delta,u}(G)$ is the discrete directional derivative of G for the direction u and step-size δ. We have proved the formula:

$$\lim_{n\to\infty} \frac{|F^n(z)|}{|D_{\delta,u}(F^n(z))|} = \delta. \tag{7.2}$$

In the case of the hypercomplex fractals generated by Equation 7.1, we will show that $D_{\delta,u}(F^n(z))$ can be approximated by the formal derivative $D = \frac{d}{dz}$ of the polynomial equation, defined iteratively. Thus,

$$F^{n+1}(z) = (F^n(z))^k + q$$

and

$$DF^{n+1} = k(F^n)D(F^n)$$

so that

$$DF^{n+1} = k^n F^n(z)F^{n-1}(z)...F(z)z.$$

We then get the estimate

$$\frac{|F^n(z)|}{|D(F^n(z))|} \approx \delta,$$

and hence by choosing an appropriate constant K,

$$\frac{1}{K}\frac{|F^n(z)|}{|D(F^n(z))|} < \delta. \tag{7.3}$$

Furthermore, for the Julia sets generated by any polynomial

$$p(z) = \sum_{k=0}^{l}\sum_{i=0}^{m} p_{i0}zp_{i1}z...zp_{ik},$$

(see Section 4.3, Equation 4.10), we still have

$$\frac{1}{K}\frac{|p^n(z)|}{|D(\overline{p}^n(z))|} < \delta, \tag{7.4}$$

where the polynomial $\overline{p}(z)$ is defined by $p(z)$ in following way,

$$\overline{p}(z) = \sum_{k=0}^{l}\sum_{i=0}^{m} |p_{i0}p_{i1}...p_{ik}|z^k.$$

For example,

$$p(z) = a_2 p_2 z^2 q_2 + a_1 p_1 z q_1 + q_0,$$

where a_1, a_2 are real positive numbers and q_0, q_1, q_2, p_1, p_2 are quaternions such that $|q_1| = |q_2| = |p_1| = |p_2| = 1$. Then

$$\overline{p}(z) = a_2 z^2 + a_1 z + q_0.$$

The discrete derivative $D_{\delta,u}(p^n(z))$ can be approximated by the formal derivative of $\overline{p}(z)$ as follows:

$$\begin{aligned}\lim_{n\to\infty} |D_{\delta,u}(p^n(z))| &= \lim_{n\to\infty} \frac{|a_2 p_2((z_n+\delta u)^2 - z_n^2)q_2 + a_1 p_1(\delta u) q_1|}{\delta} \\ &\leq 2|a_2||z_n| + |a_1| = D(\overline{p}(z_n)).\end{aligned}$$

Thus Inequality 7.4 is correct.

In this next section, we will discuss more generally the efficacy of the replacement of $D_{\delta,u}$ by the formal derivative D.

It is useful to compare this distance estimate with the distance estimates obtained in Chapter 3 (Equation 3.16) in the complex case. In that case, one writes

$$G(z) = \ln(|F(z)| + 1)$$

and uses the sequence of functions $G_n = \ln(|F^n(z)| + 1)$. This gives rise to

$$\delta \approx \frac{|G_n|}{|G_n'|} = \frac{|z_n| \ln |z_n|}{|z_n'|},$$

where $z_n = F^n(z)$. This is precisely the classical distance estimate for the complex case. Inequality 3.8 is the practical distance estimate that we use in our work. The derivation of this last estimate is sketched in the next section.

7.4 Calculating the Derivative in Higher Dimensional Space

We assume that the set

$$\overline{J}(F) = \{z \in \mathbf{R}^N \mid F^n(z) \not\to \infty \text{ as } n \to \infty\} \tag{7.5}$$

is a compact (closed and bounded) subset of $\mathbf{R}^N$. Call $\overline{J}(F)$ the filled Julia set of F as before, as opposed to its boundary

$$J(F) = \{z \in \overline{J}(F) \mid \text{ every neighborhood of } z \text{ contains points } z' \notin \overline{J}(F)\}. \tag{7.6}$$

Now recall our set-up: We have a point z that is outside $J(F)$ so that $|F^n(z)| \to \infty$ as $n \to \infty$. We wish to estimate the distance $\delta > 0$ of z from the Julia set $J(F)$. Given that δ is the distance from z to $J(F)$, we know there exists a unit vector $u \in \mathbf{R}^N$ such that $z^* = z + \delta u \in J(F)$. We then estimate via the quotient

$$\frac{|F^n(z)|}{\left|\frac{F^n(z+\delta u)-F^n(z)}{\delta}\right|} \approx \delta \tag{7.7}$$

for n large. See Figure 1.3 in Chapter 1.

Now consider the special case $F(z) = z^k + q$ where z^k is the k-th power map on $\mathbf{R}^N$ with $N \leq 8$. In this case we can consider the power map as part of the general multiplication structure on $\mathbf{R}^8$ that is given by **Cayley multiplication**:

$$(a + Jb)(c + Jd) = (ac - d\overline{b}) + J(cb + \overline{a}d).$$

Here $a, b, c, d \in \mathbf{R}^4$ are quaternions and $a + Jb$ denotes an arbitrary point in $\mathbf{R}^4 \times \mathbf{R}^4 = \mathbf{R}^8$. It is not hard to see that the k-th power mapping in the Cayley numbers is equivalent to our general power mapping restricted to these dimensions. The advantage of working "under the wing" of Cayley multiplication is that we can multiply any two elements of $\mathbf{R}^8$ and get an element of $\mathbf{R}^8$ as a result. A key property of the Cayley numbers is that $|zw| = |z||w|$ where $|z|$ denotes the length of the vector z in Euclidean space.

Lemma 34 *For $z, q \in \mathbf{R}^N$ with $N \in \{1, 2, 4, 8\}$ and $F(z) = z^k + q$, the difference quotient*

$$\left|\frac{(z + \delta u)^k - z^k}{\delta}\right|$$

is closely approximated by

$$|kz^{k-1}|,$$

and $D(|F^n|)$ can be approximated by

$$k|z|^{k-1}D(|F^{n-1}(z)|).$$

Proof: We first give the details of the proof for $k = 2$, and then use this as the base for an inductive proof for arbitrary k. Therefore, we first take

$$F(z) = z^2 + q.$$

Consider $G = F(z)$ and $D_{\delta,u}(|F^n(z)|)$. Set $z_n = F^n(z_0)$, $z_n^* = F^n(z^*)$. Then we have

$$D_{\delta,u}(|F^n(z_0)|) = \frac{|z_n - z_n^*|}{\delta}.$$

We can write

$$\frac{|z_n - z_n^*|}{\delta} = \frac{|z_n - z_n^*|}{|z_{n-1} - z_{n-1}^*|}\frac{|z_{n-1} - z_{n-1}^*|}{|z_{n-2} - z_{n-2}^*|}\cdots\frac{|z_1 - z_1^*|}{|z_0 - z_0^*|}. \tag{7.8}$$

For each $\frac{|z_j - z_j^*|}{|z_{j-1} - z_{j-1}^*|}$, $j = 1, 2, 3, ..., n$, we can evaluate,

$$\frac{|z_j - z_j^*|}{|z_{j-1} - z_{j-1}^*|} = \frac{|z_{j-1}^2 - (z_{j-1}^*)^2|}{|z_{j-1} - z_{j-1}^*|}$$

$$\leq \frac{|z_{j-1}^2 - z_{j-1}z_{j-1}^*|}{|z_{j-1} - z_{j-1}^*|} + \frac{|z_{j-1}z_{j-1}^* - (z_{j-1}^*)^2|}{|z_{j-1} - z_{j-1}^*|}$$

$$= |z_{j-1}| + |z_{j-1}^*|.$$

So

$$\frac{|z_n - z_n^*|}{\delta} \leq (|z_{n-1}| + |z_{n-1}^*|)(|z_{n-2}| + |z_{n-2}^*|) \dots (|z_0| + |z_0^*|).$$

If n is large enough, then $|z_n| \gg |z_n^*|$ and hence

$$(|z_{n-1}| + |z_{n-1}^*|)(|z_{n-2}| + |z_{n-2}^*|) \dots (|z_0| + |z_0^*|) \leq (2|z_{n-1}|)(2|z_{n-2}|) \dots (2|z_0|)$$

$$= D(|F^n(z)|) = 2|z_{n-1}|D(|F^{n-1}(z)|).$$

So, we have proved for the quadratic function $f(z) = z^2 + q$ that

$$D_{\delta,u}(|F^n(z_0)|) = \frac{|z_n - z_n^*|}{\delta} \leq D(|F^n(z)|), \tag{7.9}$$

and the distance estimate for a point outside the Julia set to the Julia set is

$$d(z, K_q) = \delta \geq \frac{|z_n|}{D(|z_n|)}, \tag{7.10}$$

where

$$D(|z_n|) = k|(z_{n-1})^{k-1}|D(|z_{n-1}|). \tag{7.11}$$

Here $k = 2$.

Now we give a proof for arbitrary k. As long as we can prove

$$\frac{|z_{n-1}^k - (z_{n-1}^*)^k|}{|z_{n-1} - z_{n-1}^*|} \leq k|(z_{n-1})^{k-1}| \tag{7.12}$$

where n is large enough so that $|z_{n-1}| > |z_{n-1}^*|$, the rest of the proof is the same as the case $k = 2$. To prove Equation 7.12, we use induction on k.

First, we already know that Equation 7.12 holds for $k = 2$. We now prove that Equation 7.12 holds for k if it holds for $k - 1$. We have

$$\frac{|z_{n-1}^k - (z_{n-1}^*)^k|}{|z_{n-1} - z_{n-1}^*|} \leq$$

$$\frac{|(z_{n-1})^k - (z_{n-1})^{k-1}z_{n-1}^*|}{|z_{n-1} - z_{n-1}^*|} + \frac{|(z_{n-1})^{k-1}z_{n-1}^* - (z_{n-1}^*)^k|}{|z_{n-1} - z_{n-1}^*|} =$$

$$|(z_{n-1})^{k-1}| + \frac{|(z_{n-1})^{k-1} - (z^*_{n-1})^{k-1}|}{|z_{n-1} - z^*_{n-1}|}|z^*_{n-1}|.$$

By the induction assumption

$$\frac{|(z_{n-1})^{k-1} - (z^*_{n-1})^{k-1}|}{|z_{n-1} - z^*_{n-1}|} \leq (k-1)|(z_{n-1})^{k-2}|.$$

Hence

$$|(z_{n-1})^{k-1}| + \frac{|(z_{n-1})^{k-1} - (z^*_{n-1})^{k-1}|}{|z_{n-1} - z^*_{n-1}|}|z^*_{n-1}| \leq$$
$$|(z_{n-1})^{k-1}| + (k-1)|(z_{n-1})^{k-2}||z^*_{n-1}| \leq$$
$$k|(z_{n-1})^{k-1}|.$$

This proves Equation 7.12, and hence the result holds for any k. □

In practical computation, for the function $F(z) = z^2 + q$, we can use

$$\delta \geq \frac{|z_n|}{(|z_{n-1}| + M)(|z_{n-2}| + M)\ldots(|z_0| + M)}, \tag{7.13}$$

where M is the radius of the smallest sphere that contains the Julia set. For other functions defined via Equation 7.1, the corresponding version of the above inequality can be formulated.

We can approximate M as follows.

Let z be a point on the Julia set such that $|z| = M$. Then,

$$M^2 = |z^2| = |F(z) - z| \leq |F(z)| + |q| \leq M + |q|$$
$$M^2 - M - |q| \leq 0,$$

so

$$M \leq \frac{1 + \sqrt{1 + 4|q|}}{2} = M'. \tag{7.14}$$

Thus we can replace M by M' in Equation 7.13.

Note that letting $G(z) = \ln(|F(z)| + 1)$, we can obtain an analogous distance estimation formula. That is,

$$\delta \geq c\frac{|z_n|\ln(|z_n|)}{D(|z_n|)},$$

where c is a positive real constant. To see this, let

$$s_n = \left(1 + \frac{|z^*_n|}{|z_n|}\right)\left(1 + \frac{|z^*_{n-1}|}{|z_{n-1}|}\right)\ldots\left(1 + \frac{|z^*_0|}{|z_0|}\right).$$

Then it is not difficult to prove that s_n and $\ln(|z_n|)/2^n$ are bounded by certain constants. We will verify this in the next section. Hence,

$$\frac{\ln(|z_n| + 1) - \ln(|z^*_n| + 1)}{\delta} \leq c_1\frac{D(|z_n|)}{|z_n|},$$

where c_1 is a positive constant, and

$$\delta \geq c\frac{|z_n|\ln(|z_n|)}{D(|z_n|)},$$

with $c = \frac{1}{c_1}$.

While we can obtain these estimates in terms of the natural logarithm, in practice more direct estimates of the form $\delta \geq \frac{|z_n|}{D(|z_n|)}$ and Equation 7.13 are just as useful.

We can now summarize this discussion by stating two theorems. The proofs of these theorems are the content of the discussion above.

Theorem 35 *The distance $d(z_0, K_q)$ between a point z_0 lying outside K_q and K_q itself satisfies the following inequality*

$$d(z_0, K_q) > c\frac{|z_n|}{D(|z_n|)}, \tag{7.15}$$

where $z_0 \in \mathbf{R}^N$, $N \in \{1, 2, 4, 8\}$ and $D(|z_n|)$ is defined in Equation 7.11. K_q is the Julia set defined by function $f(z) = z^m + q$ with z, $q \in \mathbf{R}^N$, and c is a positive constant.

Remark: Our methods obtain a lower bound, and it is this lower bound that is used in the applications. A similar statement can be made involving the logarithm, as discussed earlier.

Theorem 36 *The distance $d(z_0, K_q)$ between a point z_0 lying outside K_q and K_q itself satisfies the following inequality*

$$d(z_0, K_q) > c\frac{|z_n|}{|D(\overline{p}(z_{n-1}))|}, \tag{7.16}$$

where $z_0 \in \mathbf{R}^N$, $N \in \{1, 2, 4, 8\}$ and $D(\overline{f}(z_{n-1}))$ is a sum of derivatives of the sort given in Equation 7.11. K_q is the Julia set defined by the function $p(z) = \sum_{k=0}^{l}\sum_{i=0}^{m} p_{i0}zp_{i1}z...p_{ik-1}zp_{ik}$, $\overline{p}(z) = \sum_{k=0}^{l}\sum_{i=0}^{m} |p_{i0}p_{i1}...p_{ik}|z^k$, $z, p_{ik} \in \mathbf{R}^N$, and c is a positive constant.

7.5 Another Version of the Distance Estimation Formula

By letting $G(z) = \ln(|F(z)| + 1)$ by analogy to the formula in the complex case, we obtain a hypercomplex distance estimation formula.

Theorem 37 *The distance $d(z_0, K_q)$ between a point z_0 lying outside K_q and K_q itself satisfies the following inequality:*

$$\delta \geq c\frac{|z_n|\ln(|z_n|)}{D(|z_n|)}, \tag{7.17}$$

where c is a positive real constant.

To see this, let

$$s_n = \left(1 + \frac{|z_n^*|}{|z_n|}\right)\left(1 + \frac{|z_{n-1}^*|}{|z_{n-1}|}\right)\dots\left(1 + \frac{|z_0^*|}{|z_0|}\right). \tag{7.18}$$

In the following paragraph, we will prove that s_n and $\ln(|z_n|)/2^n$ are bounded by certain constants. Theorem 37 will follow from this.

Lemma 38 *If s_n is defined as in Equation 7.18, then*

$$s_n < c_1, \tag{7.19}$$

where c_1 is a positive constant.

Proof: Without loss of generality, let $M > 1$. Thus there exists a $k \in \mathbf{Z}^+$ such that when $n > k$,

$$M\frac{|z_n^2 + c|}{|z_n|^2} > 1,$$

and

$$1 - \frac{M}{|z_{n+1}|} > 0.$$

First we assume by induction that there exists a $c \in \mathbf{R}$, $c > 0$ such that

$$s_n \le c\left(1 - \frac{M}{|z_{n+1}|}\right).$$

If we can show that

$$s_{n+1} \le c\left(1 - \frac{M}{|z_{n+2}|}\right),$$

then it will follow by induction that s_n is bounded by a constant c_1. The following calculation proves the point.

$$\begin{aligned}
s_{n+1} &= \left(1 + \frac{M}{|z_{n+1}|}\right) s_n \\
&\le c\left(1 + \frac{M}{|z_{n+1}|}\right)\left(1 - \frac{M}{|z_{n+1}|}\right) \\
&= c\left(1 - \frac{M^2}{|z_{n+1}|^2}\right) \\
&\le c\left(1 - \frac{M}{|z_{n+1}^2 + c|}\right)\left(\frac{M|z_{n+1}^2 + c|}{|z_{n+1}|^2}\right) \\
&\le c\left(1 - \frac{M}{|z_{n+1}^2 + c|}\right) \\
&= c\left(1 - \frac{M}{z_{n+2}}\right).
\end{aligned}$$

Hence s_n is bounded by a positive constant c_1. (In fact, we can take $c_1 = c$.) □

Lemma 39 *The quantity* $\ln|z_n|/2^n$ *is bounded by a positive constant.*

Proof: First we find a positive decreasing sequence t_n such that $t_n \to 0$ when $n \to 0$ and

$$\frac{1-t_n}{1-t_{n+1}} \leq \frac{\ln|z_n^2|}{\ln|z_n|^2+|q|}. \tag{7.20}$$

Let $\delta_n = t_n - t_{n+1}$. Note that

$$\frac{1-t_n}{1-t_{n+1}} < \frac{1}{1+t_n-t_{n+1}} = \frac{1}{1+\delta_n}.$$

Note also that the inequality

$$\frac{1}{1+\delta_n} \leq \frac{\ln|z_n^2|}{\ln|z_n|^2+|q|}$$

is equivalent to the following inequality

$$\delta_n \geq \frac{\ln\left(\frac{|z_n|^2+|q|}{|z_n|^2}\right)}{\ln|z_n|^2}. \tag{7.21}$$

If we can find t_n such that the latter inequality is true, then Inequality 7.20 will hold.

We set

$$t_n = 3\frac{\ln\left(\frac{|z_n|^2+|q|}{|z_n|^2}\right)}{\ln|z_n|}.$$

Then,

$$\begin{aligned}
t_n - t_{n+1} &= 3\frac{\ln\left(\frac{|z_n|^2+|q|}{|z_n|^2}\right)}{\ln|z_n|} - 3\frac{\ln\left(\frac{|z_{n+1}|^2+|q|}{|z_{n+1}|^2}\right)}{\ln|z_{n+1}|} \\
&\geq 3\left(\frac{\ln\left(\frac{|z_n|^2+|q|}{|z_n|^2}\right)}{\ln|z_n|} - \frac{\ln\left(\frac{|z_n|^2+|q|}{|z_n|^2}\right)}{\ln|z_{n+1}|}\right) \\
&\geq 3\ln\left(\frac{|z_n|^2+|q|}{|z_n|^2}\right)\left(\frac{1}{\ln|z_n|} - \frac{1}{\ln|z_{n+1}|}\right) \\
&\geq 3\ln\left(\frac{|z_n|^2+|q|}{|z_n|^2}\right)\left(\frac{1}{\ln|z_n|} - \frac{1}{\ln|z_n|^{\frac{3}{2}}}\right).
\end{aligned}$$

Here we assume n is large such that

$$\ln|z_{n+1}| = \ln|z_n^2+q| \geq \ln|z_n|^{\frac{3}{2}}.$$

Therefore the sequence t_n satisfies

$$\begin{aligned}
t_n - t_{n+1} &\geq 3\ln\left(\frac{|z_n|^2+|q|}{|z_n|^2}\right)\left(1-\frac{2}{3}\right)\left(\frac{1}{\ln|z_n|}\right) \\
&= \ln\left(\frac{|z_n|^2+|q|}{|z_n|^2}\right)\left(\frac{1}{\ln|z_n|}\right)
\end{aligned}$$

$$\geq \ln\left(\frac{|z_n|^2+|q|}{|z_n|^2}\right)\left(\frac{1}{\ln|z_n|^2}\right).$$

It is not difficult to see that t_n is also decreasing.

Second, we can prove $\ln|z_n|/2^n$ is bounded by a positive constant c_2. Since

$$\frac{1-t_n}{1-t_{n+1}} \leq \frac{\ln|z_n^2|}{\ln|z_n|^2+|q|},$$

we have

$$(1-t_n)(\ln|z_n|^2+|q|) \leq (1-t_{n+1})\ln|z_n^2|,$$

and hence

$$(\ln|z_n|^2+|q|) \leq \frac{1-t_{n+1}}{1-t_n}\ln|z_n^2|.$$

So if we assume for n sufficiently large that

$$\frac{\ln|z_n|}{2^n} \leq c_2(1-t_n),$$

then

$$\begin{aligned}\frac{\ln|z_{n+1}|}{2^{n+1}} &\leq \frac{\ln(|z_n^2|+|q|)}{2^{n+1}}\\ &\leq \frac{(1-t_{n+1})\ln|z_n|^2}{(1-t_n)2^{n+1}}\\ &\leq \left(\frac{1-t_{n+1}}{1-t_n}\right)\left(\frac{\ln|z_n|}{2^n}\right)\\ &\leq \frac{1-t_{n+1}}{1-t_n}c_2(1-t_n)\\ &= c_2(1-t_{n+1}).\end{aligned}$$

Therefore,

$$\frac{\ln|z_n|}{2^n} \leq c_2,$$

for n sufficiently large.

Finally, we conclude that $\ln|z_n|/2^n$ is bounded by a positive constant. □

Now we can prove Theorem 37.

Proof of Theorem 37:

Let's consider

$$\frac{\ln(|z_n|+1)-\ln(|z_n^*|+1)}{\delta}$$

$$
\begin{aligned}
&= \frac{\ln(|z_n|+1)-\ln(|z_n^*|+1)}{|z_n-z_n^*|}\frac{|z_n-z_n^*|}{|z_{n-1}-z_{n-1}^*|}\dots\frac{|z_1-z_1^*|}{|z_0-z_0^*|} \\
&\leq \frac{\ln(|z_n|+1)-\ln(|z_n^*|+1)}{|z_n-z_n^*|}(|z_n|+|z_n^*|)\dots(|z_0|+|z_0^*|) \\
&= \frac{\ln(|z_n|+1)-\ln(|z_n^*|+1)}{|z_n-z_n^*|}\left(1+\frac{|z_n^*|}{|z_n|}\right)\dots\left(1+\frac{|z_0^*|}{|z_0|}\right)|z_n|\dots|z_0| \\
&= \frac{\ln(|z_n|+1)-\ln(|z_n^*|+1)}{|z_n-z_n^*|}\left(1+\frac{|z_n^*|}{|z_n|}\right)\dots\left(1+\frac{|z_0^*|}{|z_0|}\right)\frac{|D(z_n)|}{2^n} \\
&\approx \frac{\ln|z_n|}{2^n}\left(1+\frac{|z_n^*|}{|z_n|}\right)\dots\left(1+\frac{|z_0^*|}{|z_0|}\right)\frac{|D(z_n)|}{|z_n|} \\
&\leq \frac{\ln|z_n|}{2^n}\left(1+\frac{M}{|z_n|}\right)\dots\left(1+\frac{M}{|z_0|}\right)\frac{|D(z_n)|}{|z_n|}.
\end{aligned}
$$

Here M is the radius of the smallest sphere that contains the Julia set, as defined in the previous section. By Lemma 38 and Lemma 39, we have

$$\frac{\ln(|z_n|+1)-\ln(|z_n^*|+1)}{\delta} \leq c_1c_2\frac{|D(z_n)|}{|z_n|},$$

where c_1 and c_2 are positive constants. Therefore,

$$\delta \approx \frac{\ln(|z_n|+1)}{\frac{\ln(|z_n|+1)-\ln(|z_n^*|+1)}{\delta}} \geq c\frac{|z_n|(\ln|z_n|)}{|D(z_n)|}.$$

where $c=\frac{1}{c_1c_2}$. □

Part IV

Inverse Iteration, Ray Tracing and Virtual Reality

Chapter 8

Inverse Iteration: An Interactive Visualization

Inverse iteration was first used to produce Julia sets by Mandelbrot [26]. The method was further developed by Peitgen and Richter [32] and then by Peitgen and Saupe [33], who took advantage of certain properties of Julia sets to make the algorithm more efficient.

The inverse iteration method relies on the fact that a Julia set is the closure of the set of repelling periodic points [4]. If we apply the inverse mapping, the Julia set becomes attractive rather than repulsive. Since the Julia set is attractive under the inverse mapping, application of the inverse function to any point near the Julia set produces another point even closer to the Julia set.

8.1 Classical Inverse Iteration

The inverse iteration algorithm has been used extensively to visualize fractals in the complex plane. The basic algorithm given below utilizes the attractivity of the Julia set under the inverse equation. We assume that the function $f(z)$ (possibly $f(z) = z^2+c$) is being iterated. The algorithm can be described as:

Algorithm 1

```
1. Select an arbitrary initial point $z_0$.
2. For $i \in \{1, 2, 3, ..., N\}$
    (a) Let $z_i$ be chosen at random from the possible values of $f^{-1}(z_{i-1})$.
    (b) If $i > M$, then plot $z_i$ and $-z_i$.
3. End for.
```

The parameter N, the number of points generated, should be chosen so that the routine finishes in a reasonable amount of time and enough points are plotted to adequately define the set.

If the initial point is chosen at random, then several iterations of the inverse function will bring the point very close to the set. The value M may be set to zero if the initial point is chosen sufficiently close to the Julia set.

For $f(z) = z^2 + c$, the simplest version of the inverse iteration method is to randomly choose the positive or the negative square root at each stage, plotting both roots, but using only one of them to take the next square root. The result corresponds to taking a random path through the tree of all possible square roots. This method results in a simple program but leaves out many fine details in the actual set. These details correspond directly to non-random paths through the tree. By searching the entire tree structure, one can eliminate this difficulty.

Searching the tree can be accomplished without great cost to computer memory. Only a linear amount of information is necessary to search the tree down to a given depth. Algorithm 2 provides an example of a prefix traversal of the tree using a recursive function call. In the algorithm we define $g(z) = \sqrt{z - c}$ to be the positive square root in the following sense:

$$\sqrt{R(a+bu)} = \pm\sqrt{R}\left(\sqrt{(1+a)/2} + \operatorname{sign}(b)u\sqrt{(1-a)/2)}\right),$$

valid when $a^2 + b^2 = 1$, R is a non-negative real number, and u is a unit pure hypercomplex number so that $u^2 = -1$. Note that $z - c$ can be uniquely written in the form $R(a + bu)$. The variable *maxdepth* is the predetermined depth to which the tree will be searched. Note also that *return* means exit the function and return to the calling environment.

Algorithm 2

```
1. Set depth = maxdepth−1.
2. Choose initial value $z = z_0$.
3. Call plotroot(depth, $z$).
4. Definition of function plotroot(depth, $z$):
    (a) If depth= 0, return.
    (b) Plot the point $g(z)$.
    (c) Plot the point $-g(z)$.
    (d) Set $r = g(z)$.
    (e) Call plotroot(depth−1, $r$).
    (f) Call plotroot(depth−1, $-r$).
    (g) Return.
```

The inverse iteration method still has its pitfalls. Some points on the Julia set are more repulsive (inverse attractive) than others, creating regions that may be too dense or too sparse in the point-cloud image of the Julia set. However, a more refined tree search allows the inverse iteration algorithm to generate good images of complex Julia sets in real time. Methods to prune the tree structure of inverse iterates, described in Section 8.8, can even out the distribution of points in the image. Such programs are easily written on a small computer.

In the following section we will see how inverse iteration can be used to generate higher dimensional Julia sets.

8.2 Mappings in the Quaternions

In order to apply the inverse iteration algorithm, we need to know how to raise quaternions to a real fractional power.

For integer powers, such as squaring, it suffices to self-multiply a value using the quaternion multiplication rule. Note that non-commutativity does not apply to this type of multiplication since both sides of the multiplication are equal.

For the set of non-zero pure quaternions, squaring is a many-to-one mapping rather than the two-to-one mapping that would be expected. This fact was originally observed by Hamilton in [14]. It is easily demonstrated by the following calculation:

$$i^2 = (-i)^2 = j^2 = (-j)^2 = k^2 = (-k)^2 = -1,$$

which illustrates the squaring mapping to be at least six-to-one. In fact, the mapping is infinity-to-one. Let

$$S^2 = \{z \mid z \in \mathbf{H}, |z| = 1, \mathrm{Re}(z) = 0\}$$

be a unit two-sphere of pure imaginary quaternions. Then

$$z^2 \colon S^2 \longrightarrow -1.$$

In general, any such sphere of radius $r \in \mathbf{R}_+$ (the positive real axis), when squared, "collapses" to a single point in $\mathbf{R}_-$, specifically $-\sqrt{r}$.

Lemma 40 *Let $f(z) = z^2$ be the quaternion squaring function. For $r > 0$, let*

$$U_{-r} = \{q \in \mathbf{H} \mid \mathit{Re}(q) = 0, |q| = \sqrt{r}\}.$$

Then $f(U_{-r}) = -r$.

This lemma follows from the discussion above. It can also be shown that similar results obtain for any integer exponent greater than one.

8.3 The Quaternion Square Root

We can raise quaternions to fractional powers. Without loss of generality, we will only discuss fractional exponents less than one—usually called rooting. With certain exceptions, rooting is a 1-to-d mapping where d is the degree of the root. The exception, as noted above, is roots of numbers on the negative real axis, which are one-to-infinity mappings. The following lemma is equivalent to Lemma 40.

Lemma 41 *The multi-valued quaternion square root function*

$$f^{-1}(z) = \sqrt{z}$$

maps a single negative real value $-r$ to a set of points $U_{-r} = \sqrt{r}S^2$.

The root of a quaternion number may be computed in much the same way that the complex square root is calculated.

Lemma 42 *The square roots of a quaternion $q = a + bi + cj + dk$, where at least one of b, c and d is not zero and $a, b, c, d \in \mathbf{R}$, is*

$$q^{\frac{1}{2}} = l\sqrt{r}\cos\left(\frac{\theta}{2}\right) + l\sqrt{r}\sin\left(\frac{\theta}{2}\right)\left(\frac{b}{r_1}i + \frac{c}{r_1}j + \frac{d}{r_1}k\right). \qquad (8.1)$$

where $l = \pm 1$, and $\theta = \arccos\left(\frac{a}{r}\right)$, $r = \sqrt{a^2 + b^2 + c^2 + d^2}$, $r_1 = \sqrt{b^2 + c^2 + d^2}$.

Proof: We can write q as

$$q = A + Bu,$$

where $A = a, B = r_1 = \sqrt{b^2 + c^2 + d^2}$ and $u = \left(\frac{b}{r_1}i + \frac{c}{r_1}j + \frac{d}{r_1}k\right)$. So $r = \sqrt{a^2 + b^2 + c^2 + d^2} = \sqrt{A^2 + B^2}$.

Letting $\theta = \arccos\left(\frac{a}{r_1}\right)$, we can rewrite q as:

$$q = r\left(\frac{A}{r} + \frac{B}{r}u\right) = r(\cos\theta + \sin\theta u).$$

Therefore,

$$q^{\frac{1}{2}} = l\sqrt{r}\left(\cos\left(\frac{\theta}{2}\right) + l\sin\left(\frac{\theta}{2}\right)u\right), \; l = \pm 1.$$

Since $u^2 = -1$, it is not difficult to see that

$$\left(q^{\frac{1}{2}}\right)^2 = r\left(l\cos\left(\frac{\theta}{2}\right) + l\sin\left(\frac{\theta}{2}\right)u\right)^2 = r(\cos\theta + \sin\theta u) = q.$$

□

Later we will discuss how to choose efficient initial values for the inverse iteration.

One important property of the rooting function as suggested above is that null imaginaries remain null, as stated below.

Lemma 43 *If $f^{-1}(z) = \sqrt{z}$ is the inverse of the squaring map in the quaternions, then for r real and > 0, $f^{-1}(r)$ is a set of real numbers $\{+\sqrt{r},\ -\sqrt{r}\}$, where $\sqrt{r}$ denotes the real square root.*

This lemma is useful for choosing initial values for the inverse iteration algorithm.

8.4 The n-th Roots in Higher Dimensions

In this section we point out a natural generalization of the n-th root of a quaternion, which leads to an interesting method for visualizing higher dimensional Julia sets using inverse iteration.

As in Section 7.1, we can define a value in $\mathbf{R}^{N+1}$ as

$$X = a + bu,$$

or

$$X = r(\cos\theta + \sin\theta u),$$

where r is a real number, $u = (u_1, u_2, ..., u_n)$, and $0 < \theta < 360$.

Then we can simply deduce the n-th roots of X to be

$$X_l = \sqrt[n]{r}\left(\cos\frac{\theta + 2l\pi}{n} + u\sin\frac{\theta + 2l\pi}{n}\right), \tag{8.2}$$

where $l = 0, 1, ..., n-1$.

We can compute X_l^n just as in the complex numbers:

$$X_l^n = r\left(\cos\frac{\theta + 2l\pi}{n} + u\sin\frac{\theta + 2l\pi}{n}\right)^n.$$

Since $u^2 = -1$, we can treat u as the i in the complex plane when we simplify the above equation. Then we can have X_l^n in the form

$$X_l^n = R(\cos\phi + u\sin\phi).$$

Equation 8.2 can be used in the inverse iteration algorithm for higher dimensional Julia sets.

We can also easily deduce a "collapsing $(N-2)$-sphere" property for N-dimensional variables.

Lemma 44 *The inverse of the squaring function, the square root function,*

$$f(X) = \sqrt{X}$$

maps a single negative real value r to a set of points $U_r = \sqrt{-r} \subset S^{N-2}$.

8.5 Quaternion Julia Sets via Inverse Iteration

The Julia set is repulsive with respect to the polynomial $f = z^2 + q$, which means that the orbits of points near the set move farther away from it. Conversely, the Julia set is attractive with respect to the inverse function

$$f^{-1}(z) = \sqrt{z - q}, \tag{8.3}$$

because the orbits of points far away from the Julia set come closer to it under f^{-1} [32]. This means that Julia sets are a specific form of strange attractor. These shapes commonly arise from physical differential equations such as the well known Lorenz attractor.

The function used to generate the Julia sets can be any quaternion polynomial. But not all such Julia sets can be generated using the inverse iteration method. Consequently, we must first solve the quaternion polynomial $f(z) = z$ before we think about the inverse iteration. For example, it would be difficult to solve a polynomial such as:

$$p(z) = p_1 z p_2 z p_3 + p_4 z p_5 + p_6,$$

where $p_i \in \mathbf{H}$, $i = 1, ..., 6$.

However, some quaternion polynomials are good candidates for generating Julia sets by inverse iteration. For example,

$$f(z) = z^2 + p \tag{8.4}$$

and

$$f(z) = p_1 z^2 q_1 + p_0, \tag{8.5}$$

where p_1, q_1, $p_0 \in \mathbf{H}$ etc. In both of these equations, one can solve for z in terms of f and the constants.

Another problem we had mentioned earlier is the choice of initial points for a solvable polynomial function. Some points on the Julia set are more inversely attractive than others, so that some parts of the point-cloud image may appear more dense than others. We do not know a general method for choosing "good" initial values, since we are using a countable set of points to describe a Julia set that consists of uncountably many points. This is one of the disadvantages of the inverse iteration method. However, one method for producing better images is to use a multiplicity of initial points. The tree search algorithm described above and methods for tree pruning (see Section 8.8) can also improve image quality. Implementing the tree search is computationally practical for two-dimensional Julia sets but requires a great deal of computational power for higher dimensions, and one must pay particular attention to the places where the roots have infinite multiplicity.

8.6 Functions Used in the Inverse Iteration Method

Not all Julia sets generated by quaternion polynomials can be visualized by the inverse iteration method. Functions of the form

$$F(z) = c_1 z^2 + c_2, \tag{8.6}$$

with $c_1, c_2, \in \mathbf{C}$, are amenable to inverse iteration. We will see later that

$$f(z) = z^2 + q, \tag{8.7}$$

with q a quaternion, has the same dynamics as $f(z) = z^2 + c$ for an appropriate c in the complex numbers. This will be proved in Theorem 48 Section 9.8. We will describe the Julia set J_q defined by Equation 8.7 using the inverse iteration method that we used for Equation 8.6.

Since we can only observe a 4D Julia set in 3D space, we need to find some method to represent a 4D object in 3D space. The following theorem suggests how the entire Julia set defined by Equation 8.6 can be visualized using the inverse iteration method.

Theorem 45 *The dynamics of Equation 8.6 is independent of the angle ϕ in*

$$z = z_1 + e^{i\phi} z_2 j,$$

where $z_1, z_2 \in \mathbf{C}$. In particular, letting $g_\phi(z_1+z_2 j) = z_1 + e^{i\phi} z_2 j$, then $g_\phi(F(g_{-\phi}(z))) = F(z)$. We can view the function $g_\phi(z)$ as a rotation of 4-space that fixes the complex plane ($z_2 = 0$). Thus, we see that the 4D Julia set is obtained by rotating the 3D Julia set about the complex plane.

Remark: We call this property of the 4D Julia set "j-k equivalence."

Proof: Let $c_1 = a_1 + b_1 i$, $c_2 = a_2 + b_2 i$ and

$$g_\phi(z) = \mathrm{Re}(z) + \mathrm{Im}_i(z) i + e^{i\phi}(\mathrm{Im}_j(z) j + \mathrm{Im}_k(z) k)$$

be the rotation of quaternion z about the complex plane that leaves points in $\mathbf{C}$ fixed. Note that any arbitrary $z \in \mathbf{H}$ can be expressed as

$$z = z_1 + z_2 j,$$

where $z_1, z_2 \in \mathbf{C}$. We can choose ϕ to be the angle such that $e^{-i\phi} = \overline{z_2}/|z_2|$.

Notice that

$$g_{-\phi}(z) = z_1 + e^{-i\phi} z_2 j = z_1 + |z_2| j$$

takes z from $\mathbf{H}$ into $\mathbf{H}^{\frac{3}{4}}$, where $\mathbf{H}^{\frac{3}{4}}$ denotes the quaternions with zero k component. Then,

$$F(g_{-\phi}(z)) = c_1(g_{-\phi}(z))^2 + c_2$$
$$= c_1(z_1 + e^{-i\phi} z_2 j)^2 + c_2 = c_1(z_1 + |z_2| j)^2 + c_2$$
$$= c_1(z_1^2 + z_1 |z_2| j + |z_2| j z_1 - |z_2|^2) + c_2$$
$$= c_1(z_1^2 + |z_2| z_1 j + |z_2| \overline{z_1} j - |z_2|^2) + c_2$$
$$= c_1(z_1^2 - |z_2|^2) + c_2 + 2\mathrm{Re}(z_1) c_1 |z_2| j.$$

So

$$g_\phi(F(g_{-\phi}(z))) = c_1(z_1^2 - |z_2|^2) + c_2 + 2\mathrm{Re}(z_1) e^{i\phi} c_1 |z_2| j$$

$$= c_1(z_1^2 - |z_2|^2) + c_2 + 2\text{Re}(z_1)z_2c_1j,$$

which is the same result as

$$\begin{aligned} F(z) &= c_1(z_1 + z_2j)^2 + c_2 \\ &= c_1(z_1^2 + z_1z_2j + z_2jz_1 + z_2jz_2j) + c_2 \\ &= c_1(z_1^2 + z_1z_2j + \overline{z_1}z_2j + z_2\overline{z_2}j^2) + c_2 \\ &= c_1(z_1^2 - |z_2|^2) + c_2 + 2\text{Re}(z_1)z_2c_1j. \end{aligned}$$

So

$$g_\phi(F(g_{-\phi}(z))) = F(z).$$

□

Notice that the hypothesis of this theorem requires that $c \in \mathbf{C}$. We will prove in the next chapter that the quaternionic Julia set defined by $f(z) = z^2 + q$ is equivalent by rotations to the complex Julia set defined by $f(z) = z^2 + c$ for some $c \in \mathbf{C}$.

8.7 An Algorithm for the Inverse Iteration Method

The algorithm presented in this section is an extension of the classical inverse iteration algorithm given in Section 8.1. Without loss of generality, by the discussion of last section, Equation 8.6 is used to describe the algorithm. The inverse of Equation 8.6 is

$$f^{-1}(z) = \sqrt{c_1^{-1}z - c_1^{-1}c_2}. \tag{8.8}$$

When we write the square root sign, we mean that one randomly chooses either the positive or the negative square root. Improvement of the algorithm involves more than just changing the square root function. Some properties of quaternions can be used to make the algorithm more efficient.

Algorithm 3

```
1. Compute the set of initial points $I = \{z_0\}$.
2. For each point $z_0 \in I$,
    (a) for $l \in \{1, 2, ..., N\}$
        • If $c_1^{-1}(z_{l-1} - c_2) \in \mathbf{R}_- = \{r \in \mathbf{R} \mid r < 0\}$, choose $z_l$ from the circle
              $S_1 = \{z \mid |z| = \sqrt{c_1^{-1}(z_{l-1} - c_2)},\ Re(z) = Im_k(z) = 0\}$.
        • Else compute $z_l$ by equation 8.8.
    (b) Plot $z_l, -z_l, z_l - 2Im_j(z_l), -z_l + 2Im_j(z_l)$.
    (c) End for.
```

3. End for.

The property that 2-spheres are collapsed by the power mapping is used to define the initial values of the backward orbits. Since the only way to forward iterate from **H** to **C** is to apply the mapping to a quaternion whose square or higher power is a complex number, the best way to start a quaternion backward orbit is at such a point.

To choose the initial values, we can compute

$$\rho = \min_{y>0}\{y \mid yi \notin \text{ the Julia set}\}$$

by extending a ray from the origin up to the i axis to find the least positive i value not in the filled Julia set. Then by the "j-k equivalence" property, we can choose the set of initial values to be

$$I = \{z_0 \mid z_0 = \rho(\cos\theta i + \rho\sin\theta j), \theta \in [0, 2\pi)\}.$$

Since the initial values of the process are taken from a circle, this algorithm can be understood as the inverse iteration of loops. These loops, however, become quite convoluted in the deeper branches of the inverse iteration tree.

The square root returns at least two values, so both are plotted. Theorem 45 allows us to reflect each point across **C**. These two symmetries allow each point to be plotted as four in the resulting point cloud. The production of four points from each iteration reduces the number of quaternion square root function calls, which are computationally expensive.

Another point worth keeping in mind is that once $z_i \in \mathbf{C}$ for some i, then $f^{-1}(z_i)$ will remain in **C** for some time until the iteration sends it to the negative real numbers, $\mathbf{R}_-$. Therefore, should it occur that $z_i \in \mathbf{C}$, it is advisable to restart the algorithm with a different initial value.

When forward iteration is applied to a point with zero k component, the result also has zero k component. This is not true for the inverse iteration. Theorem 45 shows how any point in the Julia set with non-zero k component can be rotated into a point in the Julia set with zero k component. This can be done by setting

$$z = \text{Re}(z) + \text{Im}_i(z) + j\sqrt{(\text{Im}_j(z))^2 + (\text{Im}_k(z))^2}$$

for any z in the Julia set. This way we can describe a 4D Julia set in 3D space.

In the next section we provide a method to improve the algorithm.

8.8 Tree Pruning

We already know it is difficult to produce enough points to define the image of a Julia set in the quaternions. Since the inverse function in Equation 8.8 produces two values after each

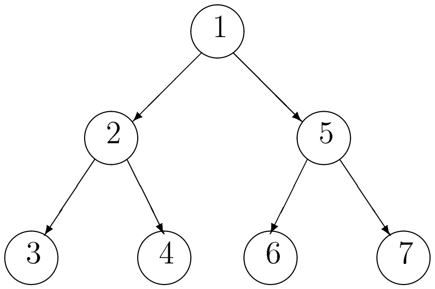

Figure 8.1: Tree Structure

iteration, we can use a tree structure for the inverse iteration of an initial point. The entire binary tree of values may be generated using only $O(\log_2 n)$ space by storing the values in the order suggested by Figure 8.1. Since the immediate descendants of a node only differ in sign, only one value needs to be stored. The tree structure also allows pruning. The pruning operation halts the inverse iteration at a certain point such that the point's descendants are not generated. This is necessary for solving the point distribution problem mentioned earlier.

There are several forms of tree pruning: temporal redundancy pruning, spatial redundancy pruning and repulsiveness pruning.

Temporal redundancy pruning is the method presented in Algorithm 3. The idea is to stop traversing a path whose root value has already been plotted. This is one of the easiest ways to prune a tree to obtain a more even distribution of values.

If every point is to be checked against all previously generated points, the algorithm would result in using the 3D object-space. Instead, the points are checked against the viewable subset of previously generated points. Although more accurate in object-space than image-space, the result is still a more even distribution. The main drawback to this technique results from the Butterfly Effect, in which the n-th iteration of a point may be arbitrarily far from the n-th iteration of its nearest neighbors. This becomes a problem, since our comparisons are only as accurate as the resolution of the image. The Butterfly Effect may be controlled by storing with each value z in the tree, the number of collisions with the set previously generated by z and its ancestors. If this number becomes too large, then the tree is pruned at this node.

Spatial redundancy pruning is derived from a paper on the ergodic property of inverse iteration [32] and was developed by Peitgen and Richter to produce some of the images illustrating their book [33]. By this method, one creates a grid of lower resolution than the final image, keeping a count of points contained in each domain of the grid. If the count reaches a maximum number, the traversal is discontinued. This seems to work well in the complex plane but is not efficient in 3D, since a three-dimensional array must be used. If the array is nearly the resolution of the resulting image, an object-space algorithm results. Smaller arrays retain the image-space complexity of the algorithm, but are not as useful.

Perhaps a reasonable compromise between size and effectiveness could be accomplished.

Another method is to prune the tree by measuring the repulsiveness (inverse attractiveness) of values. If a value is very repulsive, its derivative will be very large; less repulsive points have smaller derivatives. Thus, by keeping a running derivative at each point in the tree of inverse iterates, less attractive points may be emphasized. The absolute value of the derivative of each point in the Julia set is defined as

$$|z_n'| = 2|z_{n+1}||z_{n+1}'|, \tag{8.9}$$

where z_{n+1} is computed from z_n by the inverse function f^{-1}. We could set $z_0 = 1$. The derivative is then updated (in constant time) after each application of the inverse mapping [17].

8.9 Displaying Julia Sets

We have displayed the point clouds of Julia sets on an IRIS workstation, and we produced interactive animations in the Cave Automated Virtual Environment (CAVE) and the ImmersaDesk [36].

On our workstation, we used perspective projection together with animation, which gives a better sense of depth to the point-cloud image. Stereo was used to generate more realistic images using the following formula,

$$x' = x, \tag{8.10}$$

$$x' = x - \alpha z, \tag{8.11}$$

$$y' = y. \tag{8.12}$$

Equation 8.10 computes the left image and Equation 8.11 computes the right image. This is computed for the positive x-axis pointing to the viewer's left and positive z-axis pointing toward the viewer. This is the same as the CAVE coordinate system. The parameter α is an adjustable constant. We take $\alpha = 0.12$ if the center of focus of the stereo projection is to be at the $z = 0$ plane.

Another 3D cue is obtained by depth-cueing; see [17]. This is a technique by which the color of an object fades for points farther away as if the point-cloud is viewed through a fog. A similar depth cue uses a modular color scheme based on the coordinates. For example, if valid color indices range from 0 to 255, inclusive, then the function

$$c(z) = \lfloor 256(\mathrm{Re}(z) \bmod 1) \rfloor$$

will assign colors that vary across the point cloud in the direction of the real axis. See [17].

It is not difficult to produce renderable point clouds by inverse iteration. During inverse iteration, a z-buffer in maintained, and each resulting point is added to the z-buffer.

Then all the points in the z-buffer are rendered by computing their surface normal, and lighting is added according to the Lambertian lighting model. The surface normal of a z-buffer point may be found by following two other vectors. See [31]. Let

$$\overline{X} = (1, 0, z(x+1, y)),$$

$$\overline{Y} = (0, 1, z(x, y+1)),$$

where z is the z-buffer indexed by x and y. The surface normal may then be computed as

$$\overline{N} = \frac{\overline{X}}{|\overline{X}|} \times \frac{\overline{Y}}{|\overline{Y}|}.$$

The object can be rendered with shadows if a shadow buffer is maintained by projecting each point with respect to each light source and adding the result to a separate z-buffer dedicated to each light source.

The inverse iteration algorithm is much simpler and faster than other methods for visualization of quaternion Julia sets. It also does not require a computationally expensive frame buffer.

Chapter 9

Ray Tracing Methods by Distance Estimation

9.1 Distance Estimation via Ray Tracing

The method of ray tracing has made a tremendous impact on computer graphics since its introduction in the 1980's. Realistic three-dimensional scenes are rendered using a simple model of the way light illuminates an object. A "ray" of light is followed, or traced, from the eye of the viewer, through a pixel in the display device, and then to a point on the surface of the object to be rendered. The ray is deflected by the object, and then traced to the light sources illuminating the scene. This procedure is followed for each pixel, resulting in an image-space, image-time algorithm. Since the number of ray paths computed is proportional to the resolution of the display device, the complexity of the algorithm depends only on the complexity of the image, and not the complexity of the underlying object.

An important step in the ray tracing algorithm is finding the intersection of the ray with the object. Mathematical objects may be traced to detect their boundaries by finding the intersection of the ray with the object. The surface is rendered by allowing the ray to be deflected off to a light source. Intersections are not difficult to find for objects that can be represented by mathematical functions. However, finding the intersection of a ray with the surface of higher dimensional deterministic fractals is very difficult, if not impossible.

A naive method to ray trace a deterministic fractal would be to sample each point at a given resolution along each ray. This is not an entirely ridiculous idea, since it is the basis of the volumetric rendering algorithms in [24]. However, it would not be practical to classify each point on the ray, since we may need a large number of function iterations.

The distance estimation formula given in Chapter 3 gives us a lower bound for the distance from a point to a deterministic fractal. We can use this value as the lower bound of the distance along the ray of a point on the ray to the fractal. By this mechanism, the amount of sample points per ray is greatly reduced. Distance estimation therefore makes the ray tracing of higher dimensional deterministic fractals possible.

9.2 A Classical Ray Tracing Algorithm

Using the standard ray tracing model, at every pixel we traverse every point in the ray starting at the eye of the viewer and moving away along the ray. The algorithm is:

Algorithm 4

```
1. For each ray,
    (a) For each point on the ray,
        i. Classify the point with respect to the fractal.
        ii. If the point is interior, stop.
        iii. If the point is exterior, continue.
    (b) End for.
2. End for.
```

In the following section, we will optimize the ray tracing algorithm by using the distance estimation formula for higher dimensional fractals.

9.3 A Ray Tracing Algorithm Using Distance Estimation

The classical ray tracing algorithm provides a simple model, but it usually requires a large amount of computation. Consequently, many papers have been written that describe methods to optimize the algorithm. A method of particular interest is the use of **bounding volumes** in [39]. A bounding volume is a volume such that the computation of the intersection of a line with the volume is much faster than the computation of the intersection of the line and the object that the volume contains or "bounds." Bounding volumes are usually spheres or ellipsoids. They have been implemented on most objects, even fractals [6], [20].

Another method uses **unbounding volumes** instead of bounding volumes [16]. An unbounding volume defines a region containing absolutely no part of the object. Distance estimation produces unbounding volumes. Each time we estimate the distance of a point from the fractal, a sphere of that radius becomes an unbounding volume. This process greatly reduces the number of computations per ray.

Using the lower bound of the distance estimate, we no longer need to traverse rays in near infinitesimal increments. Instead, the algorithm leaps across each ray, incrementing by the amount of the estimated distance to the object. Since the distance estimate is the lower bound of the distance from the point to the fractal, the ray is guaranteed not to intersect the object. We have proved that the lower bound of the higher dimensional distance estimation formula is basically the same as in the complex case (see Chapter 7), hence this method may be applied to many different classes of higher dimensional deterministic fractals, including Julia sets and Mandelbrot sets.

Since we use the lower bound of the distance estimate, we get a fraction closer at each leap. The ray never intersects the surface of the object. Often, we can also give an upper bound to the distance estimate, which is a constant times the lower bound. Then, we can determine the intersection of the ray with the object when the difference of the upper bound and lower bound is less than a predetermined limit. The algorithm depends on a choice of $n \gg 1$ and a constant α chosen empirically.

Algorithm 5

```
1. For every ray extending from the eye to a pixel $p(x, y)$ on the projection plane.
    (a) Set an initial point $z_0$ on the ray.
    (b) While $z_0$ is not in the object, do
        i. Compute $|z_n|$ and $|z'_n|$ using the iterative function.
        ii. Compute the lower bound of the distance from $z_0$ to the object, $d_- = \alpha|z_n|/|z'_n|$.
        iii. Update $z_0$ to $z_0 = z_0 + \alpha\vec{m}$, where $\vec{m}$ is the unit vector of the ray.
    (c) End while.
    (d) Compute the color at pixel $p(x, y)$.
2. End for.
```

We discuss the algorithm in detail in the next few sections.

9.4 Quaternion Multiplication in the Algorithm

In this section we discuss quaternion multiplication, which we will need for the ray tracing algorithm. We can write a quaternion in form

$$q = a + bu,$$

where $u = u_1 i + u_2 j + u_3 k$ and $|u| = 1$. The square of a quaternion is

$$q^2 = a^2 - b^2 + 2abu.$$

This method requires less real number multiplication than multiplying two quaternions directly. We can also use this method to calculate higher dimensional squaring when we create higher dimensional deterministic fractals. Namely, for any value $X \in \mathbf{R}^N$, we can write

$$X = a + bu,$$

where $u \in \mathbf{R}^{N-1}$. Then the square of X is

$$X^2 = a^2 - b^2 + 2abu,$$

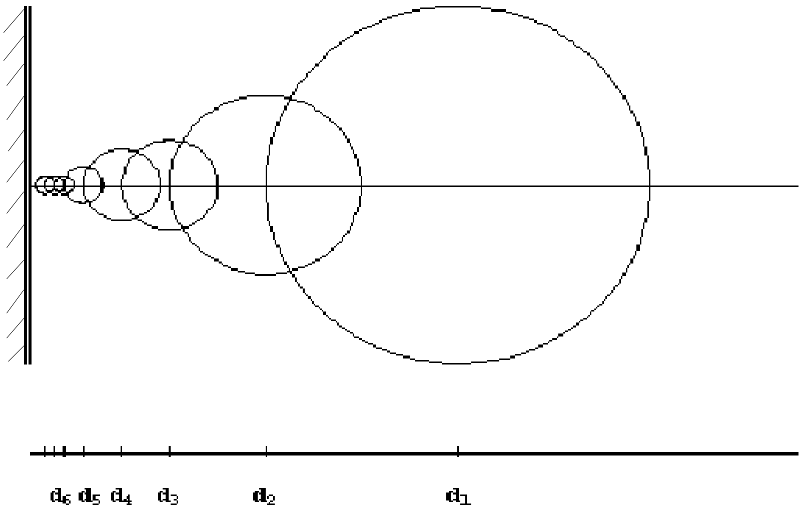

Figure 9.1: Unbounding Spheres

where we suppose $u^2 = -1$.

For any two different quaternion values $p, q \in \mathbf{H}$, we can also write

$$p = p_1 + p_2 j$$

and

$$q = q_1 + q_2 j,$$

where $p_1, p_2, q_1, q_2 \in \mathbf{C}$. Here, a quaternion is written in the form $z_1 + z_2 j$, where z_1 and z_2 are complex numbers, and the quaternion generators are 1, i, j, and $ij = k$. Thus, $(3 + 4i) + (5 + 6i)j = 3 + 4i + 5j + 6k$. Note that we identify i in the quaternions with i in the complex numbers. Then the product pq is

$$pq = (p_1 q_1 - p_2 q_2) + (p_1 q_2 - p_2 \overline{q_1})j.$$

This method is particularly useful for analyzing deterministic fractals in five or more dimensions.

9.5 Calculating the Derivative in the Algorithm

In Section 7.3, we deduced the distance estimation formula for the iterative function

$$F(z) = z^m + c, \tag{9.1}$$

which is given by:

$$\frac{1}{K} \frac{|F^n(z)|}{|D(F^n(z))|} < \delta.$$

Therefore, for any higher dimensional function of the form of Equation 9.1, the distance estimate is

$$a\frac{|z_n|}{|z'_n|} < \delta,$$

if we use z'_n to represent $D(F^n(z))$, where $a \in \mathbf{R}^+$. We can calculate z'_n by

$$|z'_n| = m|z_{n-1}|^{m-1}|z'_{n-1}|.$$

For any other quaternion functions of the form

$$F(z) = qz^m + p \tag{9.2}$$

we can deduce

$$|D_{\delta,u}(F^n(z))| \approx |D(F_1^n(z))|$$

where

$$F_1(z) = |q|z^m + |p|.$$

See Section 7.3. In fact, for any quaternion functions of the form of Equation 9.2

$$|z'_n| = m|q||z_{n-1}|^{m-1}|z'_{n-1}|.$$

For a general quaternion polynomial of the form $p(z)$ (see Section 7.3) we can introduce a new function $\overline{p}(z)$, which is

$$\overline{p}(z) = \sum_{k=0}^{l}\sum_{i=0}^{m} |p_{i0}p_{i1}...p_{ik}|z^k.$$

We can iteratively evaluate $|z'_n|$ by using $D(\overline{p}(z_n))$. That is, let

$$|z'_n| = |D(\overline{p}(z_n)||z'_{n-1}|.$$

Note that the lower bound of the distance from a point to the fractal may be much smaller and less accurate than the actual distance when we introduce the polynomial $\overline{p}(z)$. But it still provides a lower bound for the distance estimate, which we can use for the ray tracing algorithm. However, computation with $\overline{p}(z)$ may take significantly longer than Equation 9.1.

9.6 Some Important Parameters in the Algorithm

In this section, we discuss some important parameters in the ray tracing algorithm introduced in Section 9.3. First, let us look at the coefficient α in the higher dimensional distance estimation formula

$$d_- > \alpha\frac{|z_n|}{|z'_n|}. \tag{9.3}$$

Compare this with the complex distance estimation formula

$$d_- > \frac{|z_n|}{2|z'_n|}\ln|z_n|, \tag{9.4}$$

where $z_n = f^n(z_0)$ and z_0 is the initial point outside the object. After several iterations $|z_n|$ will exceed the range of numbers the computer can represent. In other words, there exists an integer n such that $|z_n|$ is greater than the maximal number the computer is capable of representing. In our tests, the limit for n was 9. Since we get a better distance estimate with larger n, we can calculate $|z_n|$ until it exceeds the maximal number of the computer. Experimentally, we find the value of $\ln|z_n|$ just before it exceeds the capacity of the computer, then treat that value as a constant. So Equation 9.4 can be approximated by

$$d_- > \frac{1}{2} a \frac{|z_n|}{|z_n'|},$$

which is essentially the same as the higher dimensional distance estimation formula in Equation 9.3.

On the other hand, since we can use $G_n(z) = \ln(|F^n(z)| + 1)$ instead of F^n, we also have a distance estimation formula for higher dimensional deterministic fractals of the form

$$d_- > \alpha \frac{|z_n|}{|z_n'|} \ln|z_n|.$$

We can therefore use Equation 9.3 as the distance estimate for all kinds of functions in all dimensions.

The parameter δ is the lower bound of the depth resolution of the object. By specializing the value δ, the clarity problem, discussed below in Section 9.11.3, can be solved. The constant δ should be large enough to support swift completion, but small enough to give adequate image detail. By setting δ to a linear or a quadratic function, certain depth cues may be synthesized. Given the updated current point $z_0^{(n)}$ on the ray defined by the initial value $z_0^{(0)}$, the ray traversal equation is

$$z_0^{(n)} = z_0^{(n-1)} + \vec{m} \min(d_-^{(n-1)}, \delta).$$

Note that the distance estimate is only accurate near the object. If the ray completely misses the object, or begins at a view point far from the object, incorrect distance estimates may result. By defining the 3D object inside a bounding sphere of radius two centered at the origin, a ray point $|z_0^{(n)}| > 2$ incremented by a positive scalar r_n is computed as

$$r_n = |z_0^{(n)}| - 2.$$

If the resulting ray point $z_0^{(n+1)}$ is not in or on the bounding sphere, the ray has missed the object.

9.7 The n-th power Family of Quaternion Mandelbrot Sets

It is known that the parameter space of complex Julia sets under the function $f(z) = z^2 + c$ is the complex Mandelbrot set. Namely, the complex Mandelbrot set is the set of constants for which the corresponding complex Julia set is connected and, as such, provides a useful index to the forms of the Julia sets. We have defined the Mandelbrot set M_4 under $f(z) = z^2 + q$, where $q \in \mathbf{H}$, as the parameter space such that zero is in the corresponding quaternion Julia set. We now turn our discussion to the Mandelbrot set M_4.

Theorem 46 *The Mandelbrot set M_4 is a surface of revolution in* $\mathbf{H}$.

Proof: For any $q \in M_4$, there is a unit quaternion p, $|p| = 1$ such that $pqp^{-1} \in \mathbf{C}$. By Lemma 28 in Chapter 6, $pqp^{-1} \in M_2$, the complex Mandelbrot set, if and only if $q \in M_4$. By Lemma 29 in Chapter 6, for any $c \in \mathbf{C}$ and any unit quaternion $p \in \mathbf{H}$, $pcp^{-1} \in M_4$. So the Mandelbrot set M_4 is the surface of revolution of M_2 in $\mathbf{H}$. □

This theorem tells us that the Mandelbrot set in quaternion space is equivalent, in some sense, to the Mandelbrot set in the complex plane. In particular, the set of magnitudes of the quaternion iteration is equal to the set of magnitudes of the complex iteration. Intuitively, the quaternionic Mandelbrot set can be viewed as a rotation of the 2D Mandelbrot set along the real axis in 4D space. A visualization method in [23] uses the look-up table and distance estimation to give a view of the 4D Mandelbrot set projected onto 3D cubes.

Using the distance estimation formula proved in Chapter 7 and the quaternion multiplication discussed in the previous section, we can view entire 4D Mandelbrot sets with the same computational complexity as the algorithms in [23]. The method can be used for all Mandelbrot sets defined by

$$f(z) = z^n + q,$$

where $q \in \mathbf{H}$. It also works well for the external Mandelbrot sets defined by

$$f(z) = p(z) + q,$$

where $p(z)$ is a fixed quaternion polynomial. The constant q is in the constant parameter space. For example, $p(z) = z^2 + z$.

The higher dimensional Mandelbrot set defined by $f(z) = z^k + c$, where $z, c \in \mathbf{R}^N$, can also be rendered using these methods. Furthermore, Theorem 46 applies to higher dimensional Mandelbrot sets, as well. Note that the theory given in Chapter 7 gives a mathematical foundation for the visualization of a great variety of high dimensional fractals, including high dimensional Mandelbrot sets.

9.8 The Quadratic Family of Julia Sets

A classical quaternion Julia set is defined by

$$f(z) = z^2 + q, \tag{9.5}$$

where $q \in \mathbf{H}$. Let us consider the Julia sets defined by Equation 9.5 with constant $q = c \in \mathbf{C}$. That is,

$$f(z) = z^2 + c, \tag{9.6}$$

is an extension of a complex function.

Using the rules of quaternion algebra, Equation 9.6 can be iterated in the quaternions, and Julia sets may be computed. Since the complex plane is a subset of the quaternions, any given complex Julia set exists in the quaternions but often has extensions outside the complex plane. In fact, if c has imaginary components, the extensions are non-trivial. The quaternionic Julia set contains more information than its complex subsets. When $c \in \mathbf{R}$, the Julia set is only a revolution of a 2D Julia set.

Theorem 47 *Any Julia set J_c defined by Equation 9.6 with $c \in \mathbf{R}$ is a surface of revolution in $\mathbf{H}$ about the real axis.*

Proof: For any initial quaternion value $Z_0 = X_0 + Y_0 u$, we can find a complex number $z_0 = x_0 + y_0 i$, where $X_0 = x_0$ and $Y_0 = y_0$, which are all real numbers. Let $\{Z_n\}, n = 0, 1, 2...$, be the orbit of Z_0, and $\{z_n\}, n = 0, 1, 2, ...$, be the orbit of z_0. Then we can prove,

$$Z_n = X_n + Y_n u,$$

$$z_n = x_n + y_n i,$$

and

$$X_n = x_n,$$

$$Y_n = y_n.$$

Actually,

$$\begin{aligned} Z_1 &= Z_0^2 + c \\ &= (X_0 + Y_0 u)^2 + c \\ &= (X_0^2 - Y_0^2 + c) + 2X_0 Y_0 u \\ &= X_1 + Y_1 u, \end{aligned}$$

and

$$\begin{aligned} z_1 &= z_0^2 + c \\ &= (x_0^2 + y_0 i)^2 + c \\ &= (x_0^2 - y_0^2 + c) + 2x_0 y_0 i) x_1 + y_1 i. \end{aligned}$$

So

$$X_1 = x_1,$$

$$Y_1 = y_1.$$

Suppose $X_{n-1} = x_{n-1}$ and $Y_{n-1} = y_{n-1}$. We can by analogy deduce $X_n = x_n$, $Y_n = y_n$. Therefore, the Julia set J_c defined by Equation 9.6 with $c \in \mathbf{R}$ is a surface of revolution in $\mathbf{H}$ about the real axis. □

A subset of the extensions of Julia sets under Equation 9.6 can be visualized in 3-space by finding the intersection of the 4D object with a 3D space spanned by 1, i, and j. And by Theorem 45, the entire Julia set generated by Equation 9.6 can be viewed in three dimensions.

For any Julia set J_q defined by the iterative function $f(z) = z^2 + q$, $q \in \mathbf{H}$, there is a complex number c such that the Julia set J_c defined by $f(z) = z^2 + c$ is equivalent to J_q by a rotation in quaternion space. We have the following theorem.

Theorem 48 *Associated with any quaternion q, there is a unit quaternion p, $|p| = 1$, and a complex number $c \in \mathbf{C}$ such that $pqp^{-1} = c$. Then the Julia set J_q defined by $f(z) = z^2+q, q \in \mathbf{H}$ can be obtained by a rotation of the complex Julia set J_c defined by $f(z) = z^2 + c$, $c \in \mathbf{C}$ in quaternion space.*

Proof: We can find the unit quaternion p and the complex number c as follows. Set $p = x + txj$, where $t, x \in \mathbf{C}$ and set $q = a + bj$, where $a, b \in \mathbf{C}$. Then by $pqp^{-1} = c$, we have $pq = cp$. That is,

$$(x + txj)(a + bj) = c(x + txj).$$

We can cancel x from this equation,

$$(1 + tj)(a + bj) = c(1 + tj).$$

So,

$$a + t\overline{a}j + bj - t\overline{b} = c + ctj.$$

That is,

$$c = a - t\overline{b} \tag{9.7}$$

$$ct = t\overline{a} + b \quad or \quad c = \frac{t\overline{a} + b}{t}. \tag{9.8}$$

Hence,

$$a - t\overline{b} = \frac{t\overline{a} + b}{t}.$$

From this equation we can solve for t, and from Equation 9.7 or Equation 9.8 we can find the complex number c. Then we can set $x \in \mathbf{C}$ such that $|x|^2 + |t|^2|x|^2 = 1$, so we can have $p = x + txj$.

According to Hamilton [14], for any quaternion q, there is a unit quaternion p and a complex number $c \in \mathbf{C}$ such that $pqp^{-1} = c$. Let $z_0 \in \mathbf{H}$ be the initial value for Equation 9.6, and let $Z_0 = pz_0p^{-1}$ be the initial value for Equation 9.5. Then

$$\begin{aligned} Z_1 = Z_0^2 + q &= (pz_0p^{-1})^2 + q \\ &= pz_0^2p^{-1} + q \\ &= p(z_0^2 + p^{-1}qp)p^{-1} \\ &= p(z_0^2 + c)p^{-1} \\ &= pz_1p^{-1}. \end{aligned}$$

Suppose $Z_{n-1} = pz_{n-1}p^{-1}$. Then we can obtain $Z_n = pz_np^{-1}$ by induction:

$$Z_n = Z_{n-1}^2 + q$$

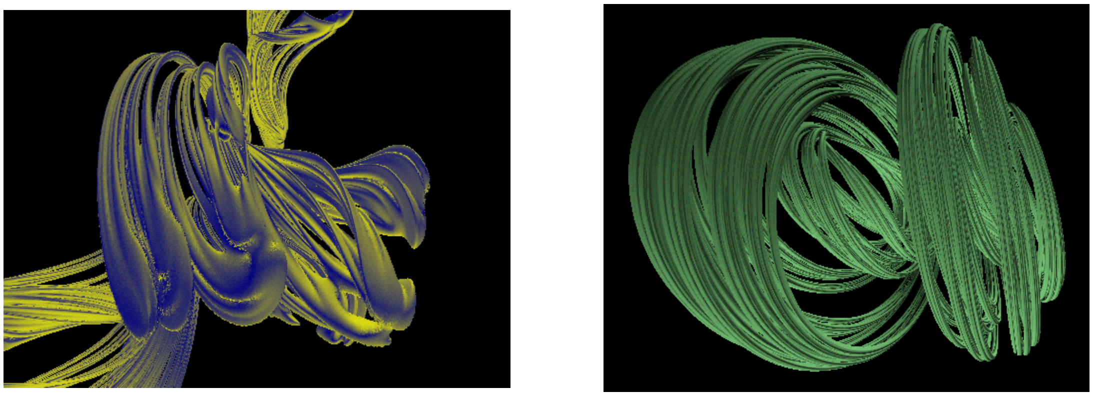

Figure 9.2: Julia sets defined by quadratic functions with different parameters c.

$$\begin{aligned}
&= (pz_{n-1}p^{-1})^2 + q \\
&= pz_{n-1}^2 p^{-1} + q \\
&= p(z_{n-1}^2 + p^{-1}qp)p^{-1} \\
&= p(z_{n-1}^2 + c)p^{-1} \\
&= pz_n p^{-1}.
\end{aligned}$$

Therefore, $z_n \in J_c$ if and only if $Z_n \in J_q$. That is,

$$J_q = pJ_cp^{-1}.$$

□

Figure 9.2 depicts ray-traced Julia sets defined by a quadratic function with different parameters c.

9.9 Generalized Quaternion Julia Sets

We can study the Julia sets that are defined by any quaternion polynomial of the form of Equation 4.10. One of the polynomials we use in the ray tracing method is

$$f(z) = pz^2 + q, \tag{9.9}$$

where $p, q \in \mathbf{H}$. This polynomial generates different quaternion Julia sets than those defined by Equation 9.5. In other words, there are some Julia sets that can be generated by a function of the form of Equation 9.9, but not by a function of the form of Equation 9.5.

We now consider the function

$$f(z) = e^{-i\theta}z^2 + e^{i\theta}c, \tag{9.10}$$

where $c \in \mathbf{C}$. This is a special case of Equation 9.9 for $p = e^{-i\theta}$ and $q = e^{i\theta}c$.

Let

$$g_\theta(z) = e^{i\theta}z \tag{9.11}$$

be a function that rotates the point z by θ counter-clockwise about the origin in $\mathbf{C}$. Another complex function can be defined as

$$f_\theta(z) = g_\theta(f(g_\theta^{-1}(z))) = e^{-i\theta}z^2 + e^{i\theta}c, \tag{9.12}$$

which causes the resulting Julia set to be rotated by θ counter-clockwise about the origin in $\mathbf{C}$.

What is perhaps unexpected is that when Equation 9.10 is iterated in quaternion space, completely different shapes occur for the same c but different θ. These shapes all share the same intersection with $\mathbf{C}$, but the topology of their extensions in $\mathbf{H}$ change dramatically. See [16]. The reason for the change is again related to the lack of commutativity of the quaternions. See Appendix A for details about these families in relation to the CD-ROM that accompanies this book.

Next, let us consider the Julia set J_1 defined by

$$f(z) = zpz + q, \tag{9.13}$$

and the Julia set J_2 defined by

$$g(z) = z^2 + pq.$$

Let $Z_0 = p^{-1}z_0$, and let $z_n = g^n(z_0)$ and $Z_n = f^n(Z_0)$. Then

$$\begin{aligned} Z_1 &= (p^{-1}z_0)p(p^{-1}z_0) + q \\ &= p^{-1}z_0^2 + q \\ &= p^{-1}(z_0^2 + pq) \\ &= p^{-1}z_1, \end{aligned}$$

and by the same reasoning $Z_n = p^{-1}z_n$, therefore,

$$J_1 = p^{-1}(J_2).$$

Thus the apparently more complex set J_1 is a rotate of the standard Julia set J_2.

In fact, any quaternion polynomial defines by iteration a Julia set. We can use a ray-tracing algorithm with the distance estimation formula to create a great variety of Julia sets for visualization. For example, we can take a cubic quaternionic function,

$$f(z) = z^3 + c.$$

The images in Figure 9.3 are quaternionic Julia sets defined by cubic functions with different parameters.

The images in Figure 9.4 are some other generalized quaternionic Julia sets.

As discussed in Chapter 7, we can also use the distance estimation formula for viewing higher dimensional Julia sets. Figure 9.5 shows three dimensional sections of five dimensional Julia sets defined by n-th power polynomials with different parameters.

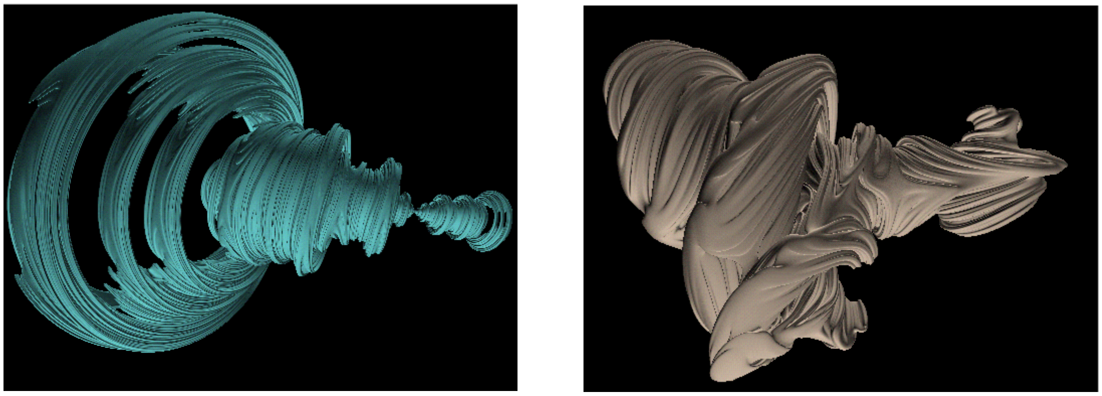

Figure 9.3: Generalized Julia sets defined by cubic quaternionic functions.

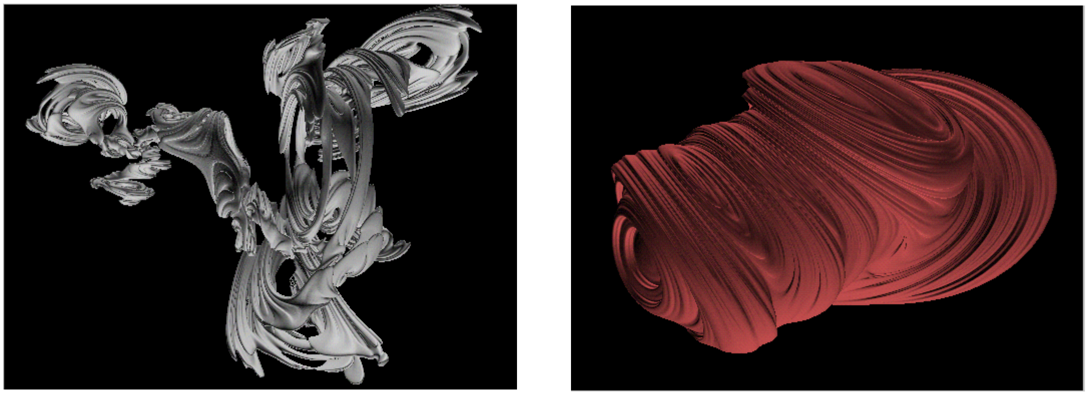

Figure 9.4: Some other generalized Julia sets.

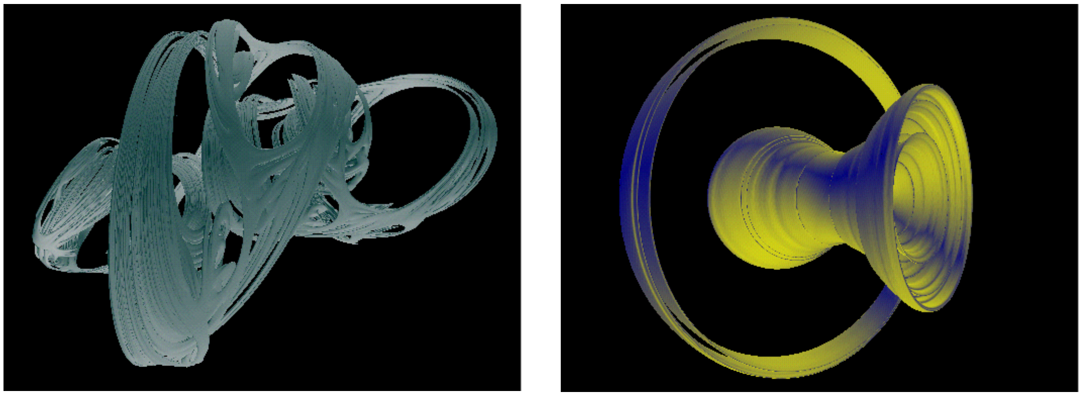

Figure 9.5: Generalized Julia sets in higher dimensional space.

9.10 Disconnected Quaternion Julia Sets

Distance estimation works well for quaternionic or higher dimensional deterministic fractals, in particular for Julia sets and Mandelbrot sets. We already know in the complex case that there are two kinds of dynamical systems under the iteration of $f(z) = z^2 + c$: connected and Cantor-like. In the quaternions, connectivity classification is more complicated. For this reason, we have not defined the quaternion Mandelbrot set as the control space of the connected Julia sets, as it is in the complex case. Our proof of the distance estimation formula in higher dimensions was based on the fact that the fractals are connected around the origin. As we change parameters to obtain disconnected hypercomplex Julia sets, the components themselves are sometimes large connected sets (not Cantor dust!). For this reason, the distance estimate can work well even in this disconnected regime.

9.11 Displaying and Rendering

Rendering is an important step in the visualization process. The object is given shading, surface texture and depth cues to create a convincing illusion of three-dimensionality. In this section we discuss rendering considerations that arise in the application of ray tracing to high dimensional deterministic fractals.

9.11.1 Light Models

When rendering a 3D image, it is essential to add light sources and to shade the surface of the object according to its position in space. The surface should appear continuous and solid. Specular reflection is commonly used to render Euclidean objects to reveal information about the material properties of the surface. However, specularity is not recommended when generating fractals, since the surface contains an infinite amount of detail at increasingly smaller scales. The extreme convolution of these surfaces scatters light in all directions, and specular lighting would suggest a diffuse, even grainy surface. Therefore, the familiar Lambertian model of diffuse shading is preferred for rendering fractals. The rendering equation we used is:

$$I(z) = K_a I_a + \sum_{i=1}^{n} K_{d,i} I_{d,i}(\overline{N}(z), \overline{L_i}(z)) \tag{9.14}$$

where $I(z)$ is the resulting color of point z. K_a is the percentage of ambient light. I_a is the color of the ambient light and therefore the color of the object. The variable n is the number of light sources. $K_{d,i}$ is the intensity of the light source, that is, the percentage of light contributed from light source i. $I_{d,i}$ is the color of light source i. $\overline{N}(z)$ is the surface normal at point z. $\overline{L_i}(z)$ is the direction of the ray of light striking point z. The only unconventional approach used in rendering the images is the addition of a light source behind the viewer to supply the ambient light contribution. This allows portions of the object that are completely shaded to convey information. This added light source is very cheap, since it casts no visible shadow, and computation of the light vector occurs when the viewing direction is computed.

9.11.2 Surface Normal

To render a fractal, we must determine a surface normal. There are several methods to compute the surface normal.

First, Norton [30] has used a normal computation that requires knowledge of the neighboring point values. He kept the entire object in accessible memory to improve efficiency, but the approach is not very useful when an image-space render-on-the-fly algorithm is required. Several other methods have been developed [15] for approximating the surface normal of a 3D fractal that require no external information.

One method for computing the surface normal uses the "neighbor cross product." If a neighboring pixel's z-values may be accessed, then it is easy to approximate the surface normal by taking the cross product of two non-colinear vectors defined by the original point and two of its neighbors. Thus, given the position in 3D of a point in a z-buffer $p_{x,y}$, the surface normal may be computed as

$$\vec{N}_{x,y} = (p_{x+1,y} - p_{x,y}) \times (p_{x,y+1} - p_{x,y}), \tag{9.15}$$

which should be subsequently normalized. Although this is a quick and well-defined method of determining the surface normal, it requires information about pixel neighbors.

Another method to find a surface normal is to use maximum distance. Pick a point z_0 on the set. Consider points z with a fixed distance from z_0. Since we have a continuous distance estimate to the Julia set, we can determine those z with maximal distance from the Julia set. A vector of the form $z - z_0$ is a good candidate for a normal to the fractal at z_0. The maximum distance normal [15] is computed as

$$\vec{N}_{z_0} = \{z - z_0 \mid \max_{z \in z_0 + \epsilon S^2} d(z)\}. \tag{9.16}$$

Thus, by maximizing the distance from a point constrained to a sphere, the vector pointing the farthest from the surface may be found.

Another method of determining the surface normal relies on the repelling nature of deterministic fractals. Points in the interior of the object are pulled toward the attractive cycle of the Julia set in its interior, if such a cycle exists. If these points take a long time to reach the cycle, then they are closer to the object. Conversely, if they approach quickly, then they must be farther away. By using these properties, we can compute the minimum iteration surface normal by:

$$\vec{N}_{z_0} = \{-(z - z_0) \mid \min_{z \in z_0 + \epsilon s^2}(n), f^n(z) \in \gamma\}, \tag{9.17}$$

where γ is the attractive cycle. The resolution of the surface normal may be increased by defining a larger maximum iteration and a more strict criterion for membership in the attractive cycle [15].

The method for determining the surface normal used in [16] and the one we favored in our ray tracing algorithm involves computing the gradient. The classical gradient as applied to the distance estimate is computed per component as:

$$N_x = d(x+\epsilon, y, z) - d(x-\epsilon, y, z), \tag{9.18}$$

$$N_y = d(x, y+\epsilon, z) - d(x, y+\epsilon, z), \tag{9.19}$$

$$N_z = d(x, y, z+\epsilon) - d(x, y, z-\epsilon). \tag{9.20}$$

The above equations define a gradient by comparing the neighbors adjacent to the faces of a cube surrounding the point in question. If the data is very convoluted, it is possible that the six-point gradient may give erroneous results. Greater refinement may be achieved by computing the eight-point gradient defined using the points touching the cube's edges, in addition to the six-point gradient, to produce a fourteen-point gradient. By defining a generalized gradient as any symmetric set of vectors from an origin to a surrounding sphere, we can implement a gradient that adaptively samples as many points as necessary to produce a sufficiently accurate surface normal. This use of the gradient is rather unconventional, since the distance estimate is not defined on the interior of a deterministic fractal. The results, however, give a reasonable justification for its use in this context, since the gradient does produce an accurately shaded, visually acceptable surface. Usually, the number of iterations required to compute the gradients is no more than the maximum number of iterations required to find the intersection of a ray with the object. Our empirical tests show that computing the surface normal using gradients is less iterative than the other methods.

9.11.3 Clarity

In projecting a three-dimensional object on a two-dimensional screen, far away parts appear smaller, and hence will receive fewer pixels for their rendering than similar parts that are closer to the viewer. Since the objects to be rendered are fractals, we should see more detail when the objects are closely inspected. However, if the minimum ray increment ϵ is a constant, the surfaces will not reveal fine structure below the level of this constant.

Clarity can be improved by selecting a clarity function $\Gamma(d)$ [16] that equals the minimum discrimination radius ϵ used by the algorithm. It is convenient to set

$$\Gamma(d) = \alpha d^{\delta}$$

where $d = |z_n - z_0|$ is the distance from the eye to the current location on the ray, and α and δ are constants described below. The clarity function determines the magnitude of the minimum ray traversal step and the size of the sphere around each point used to compute the surface normal.

The parameter δ is a depth-cueing exponent which defines how details react to distance, namely whether distant objects have more, less or equal detail when compared to near objects. If the eye is not too close to the object, that is, when $d > 1$, we have the following results: when $\delta = 0$, distant objects are shown in more detail than near objects; when $\delta = 1$, distant objects have as much detail as near objects; and when $\delta \geq 2$, distant objects have

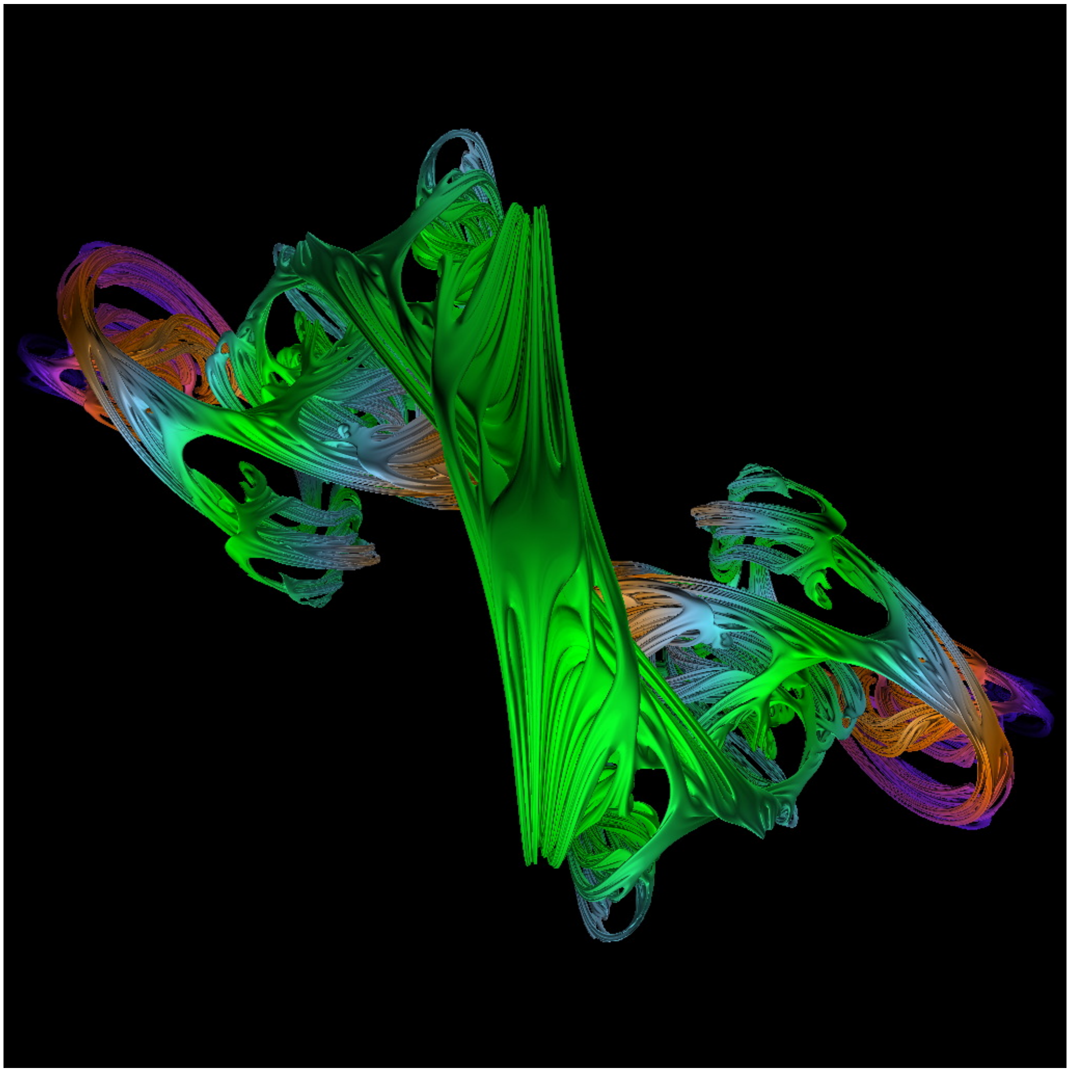

Figure 9.6: When $\delta = 0$, distant objects are shown in greater detail.

less detail than near objects. We give two examples: In Figure 9.6, $\delta = 0$ and in Figure 9.7, $\delta = 1$. With the second choice, we obtain greater clarity over a range of depths.

The parameter α is a resolution factor which should be about the size of a pixel or less. Larger α tends to blur the image and wash out the details; smaller α produces noisier results.

It should be noted that computation is much faster when the clarity function $\Gamma(d)$ is a constant. Thus, when viewing from a distance, we should choose constant clarity, the most efficient option. At higher magnifications, clarity can be adjusted to produce the desired effects.

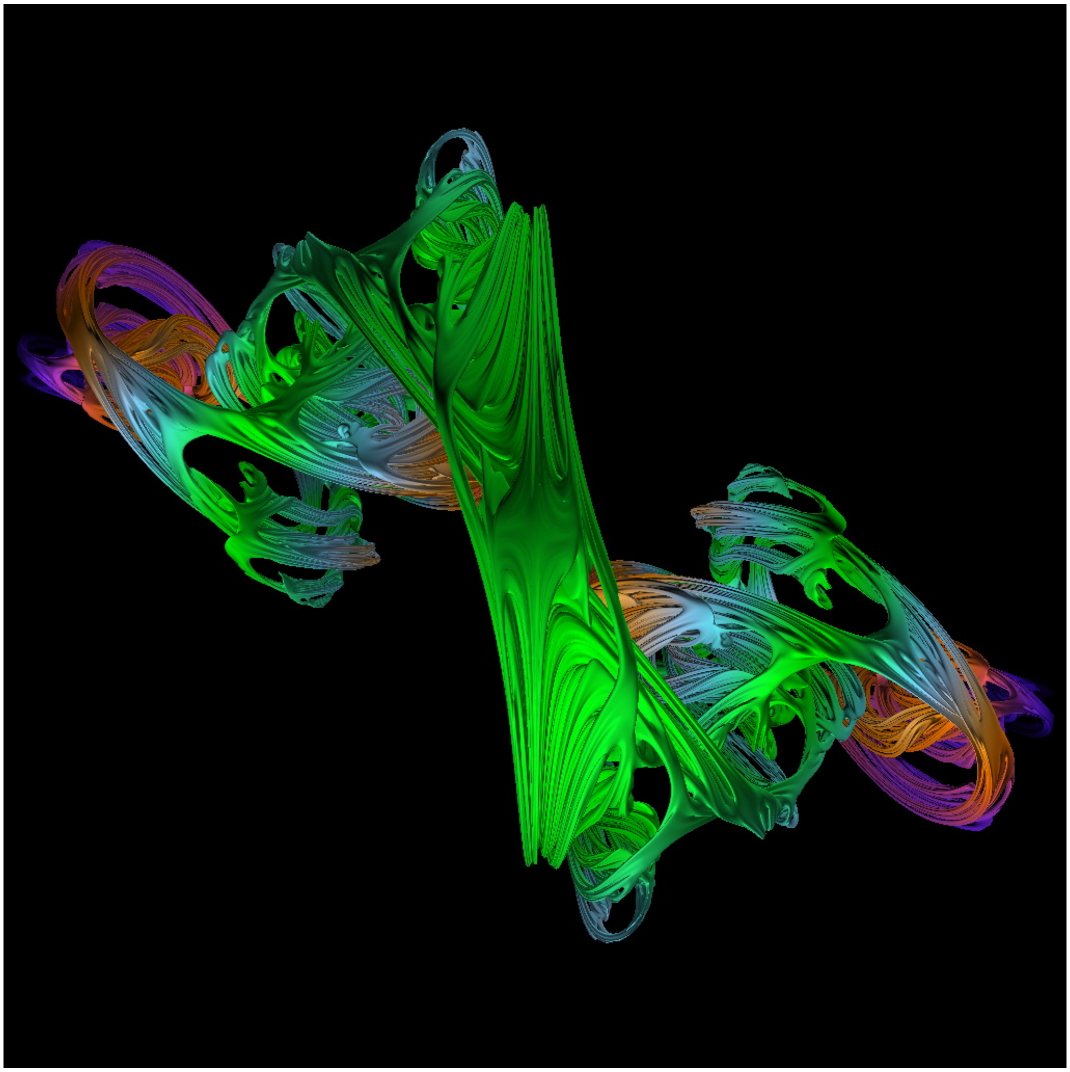

Figure 9.7: When $\delta = 1$, nearby and distant objects are shown with equal detail.

9.11.4 Other Rendering Considerations

Reflection, refraction and texture mapping may also be incorporated in the rendering of deterministic fractals. By using the surface normal to implement reflection, refraction and transparency, the fractals may be constructed of illusory chrome, crystal or clouds.

Figure 9.8 illustrates the use of the clarity function to represent the details of the Julia sets.

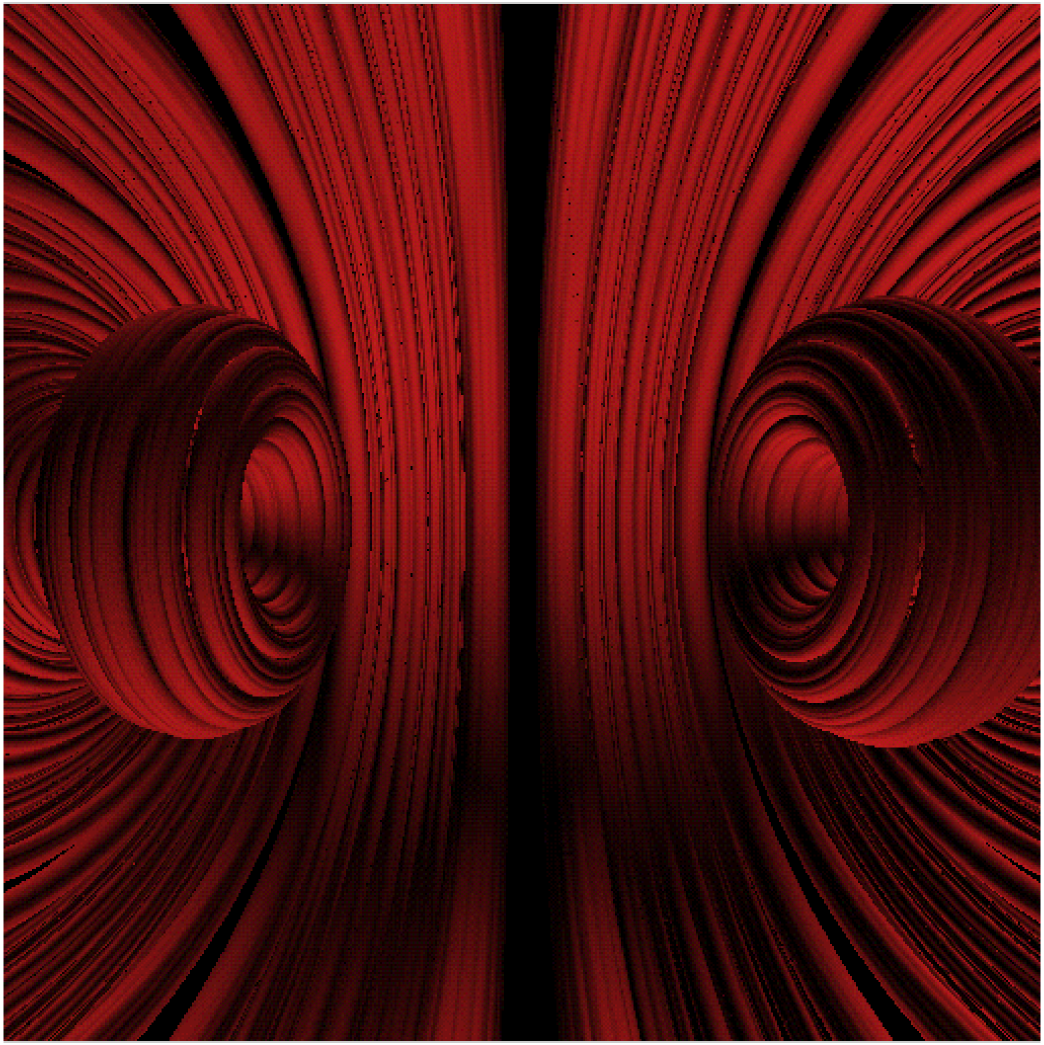

Figure 9.8: Details of a quaternionic Julia set.

Chapter 10

Quaternion Deterministic Fractals in Virtual Reality

10.1 Introduction to Virtual Reality

Virtual reality (VR) provides a way for people to visualize, manipulate and interact with computers and complex data. VR is a 3D environment of computer-generated images that exhibits the following characteristics:

1. It provides the illusion of position.
2. It provides the illusion of depth.
3. It provides for interaction with the simulated environment.

In such an environment, users can immerse themselves in simulations and interact with virtual objects in real time.

The Electronic Visualization Lab at the University of Illinois at Chicago hosts the Cave Automated Virtual Environment (CAVE), a surround-screen, surround-sound, projection-based VR system. The illusion of immersion is created by projecting 3D computer graphics into a $10' \times 10' \times 9'$ cube composed of display screens that completely surround the viewer. The CAVE is coupled with head and hand tracking systems to produce the correct stereo perspective and to isolate the position and orientation of a 3D input device. A sound system provides audio feedback. The viewer explores the virtual world by moving around inside the cube and grabbing objects with a three-button pointing device.

Multiple viewers can share a virtual experience and manipulate objects inside the CAVE, enabling researchers to collaborate in a 3D visualization context. The user holding the pointing device is the active viewer, controlling the stereo projection reference point, while the other users are passive viewers. Control of the environment can easily be transferred from one viewer to another by passing the pointer.

The rendering of higher dimensional fractals requires massive computational resources. It is nearly impossible to visualize these fractals interactively without the help of supercomputing techniques. Implementing higher dimensional deterministic fractals in the CAVE is a test case for difficult visual computations. Most importantly, the CAVE can be coupled to remote data sources, supercomputers and scientific instruments via high-speed networks.

The ImmersaDesk(TM) is a scaled-down version of the CAVE that brings 3D virtual environment technology into the office. About the size of a large drafting table, the ImmersaDesk is portable yet large enough to fill a person's field of view when he or she is seated in front of it. Images are viewed through the same lightweight stereoscopic glasses used in the CAVE.

In the next few sections, we will explain techniques that we used to visualize higher dimensional deterministic fractals in the CAVE. We expect that a good and flexible source of images of higher-dimensional deterministic fractals will raise many mathematical questions heretofore unknown. Participants can interactively change the constants specifying the fractal, evolving one fractal form from another.

10.2 Parallel Computation

Exploratory visualization of hypercomplex deterministic fractals is necessarily constrained by the long rendering times required to produce the images. In order to facilitate experimentation, our work has aimed to optimize the distance estimation algorithm and to make use of supercomputing techniques.

The algorithms had been tested on the Cray-YMP at NCSA and the Alpha-Cluster at PSC with Fortran90 and PVM in earlier experiments. By using Power C, the current implementation of the algorithms run on the SGI POWER CHALLENGE, which is a shared-memory multiprocessor (from 12 to 16) architecture based on the MIPS superscalar RISC R8000 chip with 64-bit processors.

Our code was written in IRIS PowerC, which is a parallel processing version of standard C. We used directives to mark the parallel code, and wrote the loops to be executed in parallel on multiple processors. We also used the Power C Analyzer (PCA), which can analyze serial C source code and increase the computational speed.

10.3 Data Communication

We used the CAVEcomm library developed at the Mathematics and Computer Science Division of Argonne National Laboratory to communicate between the Onyx that runs the CAVE and three separate supercomputers at NCSA that performed all the calculations.

Connections were established between the CAVE and the supercomputers, and the information that was being sent back and forth was tracked. Based on the movements of the tracker and the wand pointing device in the CAVE, information that required updating was sent to one of the three supercomputers, whichever was least busy. The supercomputer would calculate the appropriate sections of the image based on the information it received, and send the results back to the CAVE, which then updated the display.

The Information Wide Area Year (I-WAY) is an experimental high-performance network linking dozens of the country's fastest computers and advanced visualization environments. We used an I-WAY network for the data communication when the project was shown in Supercomputing '95 (SC'95). This network is based on Asynchronous Transfer Mode (ATM) technology, an emerging standard for advanced telecommunications networks. It supports both TCP/IP over ATM and direct ATM-oriented protocols. The I-WAY network provided the wide-area high-performance backbone for various experimental networking activities at SC'95. It was built from a combination of existing network connectivity and additional connectivity provided by multiple national service providers.

10.4 An Improved Display Algorithm

Displaying the Julia sets in real time was impossible, even with three supercomputers working on ray tracing the images. In order to alleviate the problem of downtime during lengthy computations, we successively refined the image. The objective was to first ray trace the entire image in lower resolution in the CAVE. We could then use the low resolution image to determine which areas needed refinement and which areas have more contrast. Areas with more contrast could be refined first, as they tend to be more visually interesting.

To refine the lower resolution image, we first need to determine the color of the current pixel. If it is not background color, it needs to be refined, otherwise we will do as follows.

Without loss of generality, let us suppose that the shape of the pixels of the lower resolution image are circles with radius r. Let D_0 denote the center of the current pixel on the projection plane. Let the eye's position be E. Let $CN = (D_0, E)$ denote the cone constructed by E and D_0. Then we can find a sequence of points on the ray from E to the center of D_0 by the following method:

We can give the lower bound of the distance d_1 from point $E_0 = E$ to the Julia set by the distance estimation formula. Let $E_1 = E_0 + d_1 u$, where u is the unit vector of the ray (from the eye to the pixel). Then we can give the lower bound of the distance d_2 from point E_1 to the Julia set by the distance estimation formula. Let $E_2 = E_1 + d_2 u$. Continue until we find some n such that $d_1 + d_2 + \ldots + d_n$ is greater than the distance from E to the center of D_0. Now we have a sequence of positive numbers $\{d_1, d_2, \ldots, d_n\}$. Let S_i, $i = 1, 2, \ldots, n$, denote the sphere with center E_i and radius d_i. If the cone $CN \subset S_1 \cup S_2 \cup \ldots \cup S_n$, then we can conclude that this pixel does not need to be refined since none of the points in the cone CN is in the Julia set.

We can give an algorithm based on the above discussion.

Algorithm 6

1. *For each pixel P on the current lower resolution projection plane:*

2. *Let P be a circle of radius r (this is the pixel shape).*
3. *If the color of the pixel is NOT a background color return YES, otherwise goto next step.*
4. (a) *Find the sequence of positive numbers $\{d_1, d_2, \ldots, d_n\}$ and the sequence of spheres $\{S_1, S_2, \ldots, S_n\}$ by using the method described above.*

   (b) *If $CN \subset S_1 \cup S_2 \cup \ldots \cup S_n$, then return NO, otherwise return YES.*
5. *End for.*

10.5 Display of Quaternion Deterministic Fractals in VR

In the preceding chapters we described two methods for the visualization of higher dimensional deterministic fractals. One method is inverse iteration, which gives a point cloud image of a Julia set. It is a real-time interactive method. The other is the unbounding-volume ray-tracing method using the distance estimation formula, which gives a finer rendering of the object. Combining these two methods, we obtain good results for displaying higher dimensional Julia sets in a VR system such as the CAVE.

First, we can input parameters which represent the Julia set we wish to see using the wand pointing device. Then, interactively, we obtain the point cloud of the Julia set by employing the inverse iteration method. The computations are performed on local computers or supercomputers (we used local SGI or SGI POWER CHALLENGE). We can adjust our position, the object's position or parameters that influence the shapes of the Julia sets. Once we obtain a view of the object that we wish to see in detail, we can send the current parameters to remote supercomputers, which perform low resolution ray tracing computations. The results from the supercomputers are obtained quickly and sent back to the CAVE for display. In the mean time, the information from the lower resolution image is sent back to the supercomputers for refinement of the image. After each computation for each pixel in the lower resolution image, the results are sent back and displayed in the proper position in the CAVE. This refining procedure can take a few minutes, during which the viewers observe the updating of the refined images. The more interesting parts are refined first, as discussed in the previous sections.

The viewer who wears the tracker can stop the procedure at any time and search through point clouds of other Julia sets. The rendering procedure will then be repeated.

10.6 Conclusion

We have outlined the visualization of higher dimensional Julia sets in virtual reality. These abstract sets of points become vivid geometrical realities in virtual space. Even with the present level of our algorithms, there is much left for further exploration. With the inevitable improvements in technology and mathematics, future visualizations will yield new insights into the structure of these fractals and the recursions that underlie them.

Appendix A

The CD-ROM that accompanies this book exhibits many families of quaternionic Julia sets in three dimensional space. This appendix gives the equations and parameterizations for these families.

Our images are produced by choosing a specific value of θ for each family, and then varying a complex constant $c = R + I$. Thus, a given family depends upon a 2-dimensional array of parameters (R, I). The function that is iterated for a fixed θ (and fixed $c = R + iI$) is

$$f_\theta = e^{-i\theta} z^2 + e^{i\theta} c$$

where z is a quaternionic variable.

The Julia set J_θ is the set of z in 4-space that do not escape to infinity under this repeated iteration. The Julia set

$$\overline{J_\theta} = J_\theta \cap \mathbf{R}^3 = \{(z_0, z_1, z_2) \mid (z_0, z_1, z_2, z_3) \leftrightarrow z_0 + iz_1 + jz_2 + kz_3 \in J_\theta\}$$

is the set that is displayed graphically.

For the record, the recursion has the following form when written out in detail:
Let $z = z_0 + iz_1 + jz_2 + kz_3$, and let $z^2 = a + bi + cj + dk$. Thus

$$a = z_0^2 - z_1^2 - z_2^2 - z_3^2$$

$$b = 2z_0 z_1$$

$$c = 2z_0 z_2$$

$$d = 2z_0 z_3.$$

Then

$$f_\theta = (\cos(\theta) - i\sin(\theta))(a + bi + cj + dk) + (\cos(\theta) + i\sin(\theta))(c_0 + ic_1).$$

Thus

$$\begin{gathered} f_\theta = \cos(\theta)a + \sin(\theta)b + \cos(\theta)c_0 - \sin(\theta)c_1 \\ +(-\sin(\theta)a + \cos(\theta)b + \cos(\theta)c_1 + \sin(\theta)c_0)i \\ +(\cos(\theta)c + \sin(\theta)d)j + (\cos(\theta)d - \sin(\theta)c)k. \end{gathered}$$

The specific distance estimation algorithm is

$$D \approx \frac{|z_n|}{|z_n'|} \ln(z_n)$$

where $z_n = f_\theta^n(z_{initial})$ and $|z_{n+1}'| = 2|z_n||z_n'|$.

Note that since $z_{n+1} = e^{-i\theta} z_n^2 + e^{i\theta} c$, then $z_{n+1}' = 2e^{-i\theta} z_n z_n'$ (by the chain rule), and so $|z_{n+1}'| = 2|z_n||z_n'|$.

Appendix B

```
PRINT"Iterated 3d Power Map - QUATERNION SLICE"
PRINT"COPYRIGHT-1986-LOU KAUFFMAN-KNOTS INC."
mag=200:INPUT"magnification";mag
'SEED input a+bu :  u=cos(theta)+isin(theta) <theta degrees>"
'formally uu=-1"

PI=3.141592653588:S=1:COUNT=1
INPUT"a";A:INPUT"b";B:INPUT"theta";T
T=2*PI*T/360
X=a:Y=b*COS(T):z=b*SIN(T)

PRINT"input CONSTANT A+BU"
INPUT"A";K:INPUT"B";L:INPUT"phi";F
F=2*PI*F/360
a=K:b=L*COS(F):C=L*SIN(F)
CLS

'now compute square roots
100 '(x,y,z) is seed., (a,b,c) is constant
X=X+a:Y=Y+b:z=z+C
R=SQR(X*X+Y*Y+z*z)
X=X/R:Y=Y/R:z=z/R:R=SQR(R)
M=SQR(Y*Y+z*z)
E=SQR(.5*(1+X)):G=SQR(.5*(1-X))
IF M=0 THEN X=R:Y=0:z=0:GOTO 180
'this case is (pos) sqr(real)
X=R*E:Y=R*Y*G/M:z=R*z*G/M
180 'end of square rooting!

COUNT=COUNT+1:IF COUNT<30 THEN 375
'now project xx=x,yy=y+.3*z
XX=X*mag+300:YY=175-(Y+.3*z)*mag
IF XX<0 OR XX>640 THEN 250
IF YY<0 OR YY> 420 THEN 250
PSET(XX,YY)
```

```
250 XX=-X*mag+300:YY=175+(Y+.3*z)*mag
IF XX<0 OR XX>640 THEN 375
IF YY<0 OR YY>420 THEN 375
PSET(XX,YY)
375 'end of screen print!

IF RND(1)>.5 THEN S=-S
X=S*X:Y=S*Y:z=S*z
GOTO 100
END
```

Bibliography

[1] Barnsley, M. F. (1988) *Fractals Everywhere*, Academic Press.

[2] Barr, A. H. (1986) "Ray tracing deformed surfaces", *Computer Graphics* **20**, (4), 287-296.

[3] Branner, B. and Hubbard, J. H. (1988) "The iteration of cubic polynomials, Part I: The global topology of the parameter space", *Acta Mathematica* **160**, (3), 143-206.

[4] Blanchard, P. (1984) "Complex analytic dynamics on the Riemann sphere", *Bulletin of the American Mathematics Society* **11**, 85-141.

[5] Blanchard P. (1986) "Disconnected Julia sets", in M. Barnsley and S. Demko, eds. *Chaotic Dynamics and Fractals*, Academic Press, Inc. 181-201.

[6] Bouville, C. (1985) "Bounding ellipsoid for ray-fractal intersection", *Computer Graphics* **19**, (3), 45-51.

[7] Bedding, S. and Briggs, K. (1995) "Regularly iterable linear quaternion maps", Submitted to *J Aust Math Soc*, April.

[8] Bedding, S. and Briggs, K. (1996) "Iteration of quaternion functions", *American Mathematical Monthly* **103**, (8), 654-664.

[9] Bedding, S. and Briggs, K. (1995) "Iteration of quaternion maps", *International Journal of Bifurcation and Chaos* **5**, (3), 877-881.

[10] Charles, J. J. (1905) *A Manual of Quaternions*, Macmillan and Co.

[11] Dang, Y. and Kauffman, L. H. (1997) "Hypercomplex fractal distance estimation", in M. M. Novak and T. G. Dewey, eds. *Fractal Frontiers*, World Scientific. 117-130.

[12] Deavours, C. A. (1973) "The quaternion calculus", *American Math. Monthly* **80**, 995-1008.

[13] Douady, A. and Hubbard, J. H. (1982) "Iteration des polynomes quadratiques complexes", *CRAS Paris* **294**, 123-126.

[14] Hamilton, W. R. (1969) *Elements of Quaternions, 3rd ed.*, Chelsea Publishing Company.

[15] Hart, J. C., Sandin, D. J. and Kauffman, L. H. (1989) "Ray tracing deterministic 3-D fractals", *Computer Graphics(SIGGRAPH '89 Proceedings)* **23**, (3), 289-296.

[16] Hart, J. C. (1989) *Image space algorithms for visualizing quaternion Julia sets*, Masters Thesis, University of Illinois at Chicago.

[17] Hart, J. C., Sandin, D. J. and Kauffman, L. H. (1990) "Interactive visualization of quaternion Julia sets", *Proceedings of Visualization '90*, IEEE Computer Society, 209-218.

[18] Holbrook, J. A. R. (1983) "Quaternionic asteroids and starfields", *Applied Mathematical Notes* **8**, (2), 1-34.

[19] Holbrook, J. A. R. (1987) "Quaternionic Fatou-Julia sets", *Annals of Science and Math Quebec* **1**, 79-94.

[20] Kajiya, T. J. (1983) "New techniques for ray tracing procedurally defined objects", *Computer Graphics(SIGGRAPH '83 Proceedings)* **17**, (3), 91-102.

[21] Kauffman, L. H. (1991, 1993, 2001) *Knots and Physics*, World Scientific.

[22] Kelley, J. (1975) *General Topology*, Springer-Verlag.

[23] Ke, Y. and Pandurange, E. S. (1990) *A Journey into the Fourth Dimension*, IEEE.

[24] Levoy, M. (1988) "Display of surface from volume data", *IEEE Computer Graphics and Applications* **8**, (3), 29-37.

[25] Mandelbrot, B. B. (1980) "Fractal aspects of the iteration of $z \rightarrow \lambda z(1-z)$ for complex λ and z", *Annals of the New York Academy of Sciences* **357**, 249-259.

[26] Mandelbrot, B. B. and Ness, J. W. (1968) "Fractional Brownian motions, fractal noise and application", *SIAM Review* **10**, 422-437.

[27] Mandelbrot, B. B. (1982) *The Fractal Geometry of Nature, 2nd ed.*, Freeman.

[28] Milnor, J. (1989) "Computers in geometry and topology", in M. Tangora, ed. *Self-Similarity and Hairiness in the Mandelbrot Set: Lecture Notes in Pure and Applied Mathematics* **114**, Marcel Dekker, 211-257.

[29] Norton, V. A. (1986) "Generation and rendering of geometric fractals in 3-D", *Computer Graphics* **16**, (3), 61-67.

[30] Norton, V. A. (1989) "Julia sets in the quaternions", *Computer Graphics* **13**, (2), 267-278.

[31] Norton, V. A., Rockword A. P. and Skolmoski, P. T. (1982) "Clamping: a method of antialiasing textured surfaces by bandwidth limiting in object space", *Computer Graphics* **16**, (3), 1-8.

[32] Peitgen, H. and Richter, P. H. (1986) *The Beauty of Fractals: Images of Complex Dynamical Systems*, Springer-Verlag.

[33] Peitgen, H. (1988) *The Science of Fractal Images* , Springer-Verlag.

[34] Perlin, K. (1985) “An image synthesizer”, *Computer Graphics* **19**, (3), 287-296.

[35] Sandin, D. J., Hart J. C. and Kauffman, L. H. (1990) “Interactive visualization of complex, stacked and quaternion Julia sets”, *Proceedings of Ausgraph’90.*

[36] Sandin, D. J., Dang. Y., Insley, J. and Kauffman L. H. (1995) “Quaternion Julia sets in virtual reality”, *Supercomputing ’95*, San Diego.

[37] Sudbery, A. (1979) “Quaternionic analysis”, *Math. Proc. Camb. Phil. Soc.* **85**, (2), 199-224.

[38] Voss, R. F. (1988) “Fractals in nature: from characterization to simulation”, in H. Peitgen and D. Saupe, eds. *The Science of Fractal Images*, Springer-Verlag.

[39] Whitted, T. (1980) “An improved illumination model for displays”, *Communications of the ACM* **23**, (6), 343-349.