James Gleick's CHAOS: The Software™

User Guide

The CHAOS software can be obtained under free Gnu license from
www.github.com/rudyrucker/chaos/
or
www.rudyrucker.com/oldhomepage/chaos.htm

Autodesk, Inc.

Publication CH1RM-02-01

May 14, 1991

AUTODESK, INC. MAKES NO WARRANTY, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, REGARDING THESE MATERIALS AND MAKES SUCH MATERIALS AVAILABLE SOLELY ON AN "AS-IS" BASIS.

IN NO EVENT SHALL AUTODESK, INC. BE LIABLE TO ANYONE FOR SPECIAL, COLLATERAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH OR ARISING OUT OF PURCHASE OR USE OF THESE MATERIALS. THE SOLE AND EXCLUSIVE LIABILITY TO AUTODESK, INC., REGARDLESS OF THE FORM OF ACTION, SHALL NOT EXCEED THE PURCHASE PRICE OF THE MATERIALS DESCRIBED HEREIN.

For conditions of use and permission to use these materials for publication in other than the English language, contact Autodesk, Inc.

Autodesk, Inc. reserves the right to revise and improve its products as it sees fit. This publication describes the state of this product at the time of its publication, and may not reflect the product at all times in the future.

Autodesk Trademarks

The following trademarks are registered in the U.S. Patent and Trademark Office by Autodesk, Inc.: ADI, ATC, Autodesk, Autodesk Animator, the Autodesk logo, AutoCAD, AutoCAD Training Center, AutoLISP, AutoShade, AutoSketch, and AutoSolid.

The following are trademarks of Autodesk, Inc.: ACAD, Advanced Modeling Extension, Advanced User Interface, AME, ATLAST, AUI, AutoCAD Development System, Autodesk Animator Clips, Autodesk Animator Theatre, Autodesk Cyberspace, Autodesk Device Interface, Autodesk Software Developer's Kit, Autodesk Training Center, AutoFlix, CA Lab, DXF, FLI, James Gleick's CHAOS: The Software, and 3D Studio.

Third Party Trademarks

All other brand and product names are trademarks or registered trademarks of their respective companies.

Program Credits

The installation program used to install James Gleick's CHAOS: The Software, INSTALL, is licensed software provided by Knowledge Dynamics Corporation, Highway Contract 4 Box 185-H, Canyon Lake, Texas 78133-3508 (USA), 1-512-964-3994. INSTALL is Copyright © 1987-1989 by Knowledge Dynamics Corporation, which reserves all copyright protection worldwide. INSTALL is provided to you for the exclusive purpose of installing CHAOS.

License Agreements

This product contains licensed program materials of Metagraphics Software Corporation, © 1986-1989 Metagraphics Software Corporation, Scotts Valley, CA 95066.

GOVERNMENT USE

Use, duplication, or disclosure by the U. S. Government is subject to restrictions as set forth in FAR 52.227-19 (Commercial Computer Software-Restricted Rights) and DFAR 252.227-7013(c) (1) (ii) (Rights in Technical Data and Computer Software), as applicable.

Excerpts

Excerpts from *Chaos: Making a New Science* are copyrighted by James Gleick (Viking Penguin 1987) and have been used with permission of the author.

3 4 5 6 7 8 9 10 CO

Table of Contents

Chapter 2 The Mandelbrot Sets 19

Chapter 3 Magnets and Pendulum *(continued)*

Chapter 4 Strange Attractors 81

Chapter 4 Strange Attractors *(continued)*

Chapter 5 The Chaos Game 121

Chapter 5 The Chaos Game *(continued)*

Chapter 6 Fractal Forgeries

Chapter 6 Fractal Forgeries *(continued)*

Chapter 7 Toy Universes 177

Chapter 7 Toy Universes *(continued)*

Chapter 8 Saving Files and Using Color Options 209

Chapter 8 Saving Files and Using Color Options *(continued)*

About This Manual

This guide tells you how to use CHAOS: The Software to explore and demonstrate aspects of chaos theory.

- The Introduction includes James Gleick's introduction to CHAOS: The Software and biographical information about James Gleick.
- Chapter 1 tells you how to install the software and use menus, buttons, and fields to work with the CHAOS programs. It's important for you to become familiar with the concepts covered in chapter 1 before continuing with the chapters on CHAOS programs.
- Chapters 2 through 7 are specific guides to each of the six programs in CHAOS. You can use the CHAOS programs and their corresponding chapters in any order.

  Each chapter begins with James Gleick's introduction to the chaos theory demonstrated by the program and his suggestions for "Things To Try" with the program. Next, you explore the program with step-by-step instructions. In later sections, you read about each feature and option of the program. Each chapter concludes with Rudy Rucker's discussion of the mathematics of that particular program.
- Chapter 8 describes how to save and load files and how to use color options—features common to most of the six CHAOS programs. Each chapter refers you to chapter 8 for this information.
- At the end of the book, you'll find "The Genesis of CHAOS: The Software" by Rudy Rucker and biographical information about Rudy Rucker. You'll also find a bibliography on chaos theory and an index.

Conventions Used in This Manual

This manual uses the following typographical conventions:

N — Keys you press on the keyboard are represented by symbols that look like the top of the key. For example, the symbol to the left represents the n key.

Shift + N — When you see two key symbols with a + symbol between them, you press both keys *at the same time*. For example, the symbols to the left mean you hold down the Shift key while you press the N key.

Keyboard shortcut: A

In addition to using the mouse and on-screen buttons to select options and features in CHAOS programs, you can also use keyboard shortcuts. Keyboard shortcuts appear below section titles. For example, in The Chaos Game program, you see ***Keyboard shortcut:*** A below the section title "Adding Maps." You can press A as the shortcut for adding maps.

↵ — This symbol represents the Return (or Enter) key on your keyboard.

cd \chaos — Text you enter appears in boldface type as shown in this example.

[redacted]

[redacted]

[redacted]

[redacted]

Introduction

"Probably the main reason these ideas are only now being discovered is that the style of exploration is entirely modern," Douglas R. Hofstadter wrote a decade ago. "The digital computer plays the role of Magellan's ships, the astronomer's telescope, and the physicist's accelerator."

Those were almost the first words I ever read about chaos, then a virtually unknown movement in science beginning to uncover strange and wonderful patterns that underlie the apparent disorder in nature. Yet even today, the extraordinary role played by computers in the birth of the science of chaos often goes unappreciated.

Computers alone were not enough. What finally broke through the traditional barriers to understanding turbulence, wildness, erratic flows of all kinds was a new style of using computers. Instead of treating these big electronic monsters as problem-solving devices, a few scientists started to use them as tools for exploration. Instead of feeding in gigantic problems on punch cards and waiting for a solution to come out, a few scientists began simply to tinker, to fiddle, to play in areas where it was not even clear exactly how to specify the problem.

This new style of exploring via computer required one thing above all: graphics. Again and again in the young history of this new science, computer graphics played a key role in the process of discovery. Researchers were able to see patterns that would have remained hidden in the numbers on their sheaves of printout. And they were able to make slight changes in key parameters—the fertility rate of a wildlife population, or the driving energy in a turbulent fluid—and watch as new patterns emerged before their eyes.

The kind of computing power available to just a few scientists a decade ago has now arrived on everyone's desktop. The set of programs collected here are not watered-down approximations, but fully realized versions of the programs that gave rise to the most important discoveries of the science of chaos.

A word that is often heard in late-night discussions of the new science is intuition. An appreciation of chaos changes one's intuition about the world—one's sense of the orderly and disorderly possibilities lurking in any complex system.

It is my hope that people who use these programs, who explore the strange worlds waiting within them, will experience this change in intuition. I believe that interactive, hands-on exploration reveals the magic of chaos in ways that neither photographs nor even motion pictures can match.

To begin to understand fractal images, Mandlebrot sets, strange attractors, and the other strange beasts in the chaotic menagerie—and to understand something about how they were discovered—it is necessary, first of all, to play.

James Gleick
11 October 1990

Introducing James Gleick

James Gleick, a former editor and reporter for the *New York Times*, is the author of *Chaos: Making a New Science* (Viking Penquin). *Chaos* won a 1987 National Book Award nomination in nonfiction and has been translated into sixteen foreign languages. Gleick is a native New Yorker and a graduate of Harvard College. His current project is a biography of the physicist Richard Feynman.

Chapter 1

Getting Started with CHAOS: The Software

This chapter tells you what equipment you need to run CHAOS: The Software, how to install the software, how to start the programs, and how to work with menus, buttons, and entry fields using either a mouse or the keyboard. The chapters on individual CHAOS programs assume you've read chapter 1 and know the basics of working with the programs.

Required Equipment

CHAOS: The Software requires the following minimum equipment:

- IBM® PC/XT, PC/AT®, 100-percent-compatible personal computer, or an IBM Personal System/2®
- Dual floppy disk drives or a hard disk and single floppy disk drive
- PC-DOS® 2.0 or later
- 640K of memory
- IBM Enhanced Graphics Adapter (EGA) and an Enhanced Graphics Monitor or IBM Video Graphics Array (VGA) with color monitor

Note: CHAOS should work without conflict with most device drivers and terminate-and-stay-resident (TSR) programs. However, if you notice problems while running CHAOS, try removing any TSRs or device drivers and reboot the computer.

Optional Equipment

The following equipment is not required, but is recommended:

- Microsoft®-compatible mouse
- 8087, 80287, or 80387 math coprocessor

The CHAOS Package

Your CHAOS package includes the following items:

- This manual
- Quick Reference Card
- CHAOS program diskettes
- License Agreement
- Registration Card

Installing CHAOS on a Hard Disk

CHAOS: The Software comes with an installation program that prompts you as you install CHAOS.

Follow these steps to use the Install program:

1. Place disk 1 in your floppy disk drive and close the drive door.
2. Make that floppy disk drive the current one. For example, if you are in the root directory on drive C and disk 1 is in drive A, enter the following:

   ```
   a: ↵
   ```

3. Start the installation program by entering this command:

   ```
   install ↵
   ```

4. Follow the on-screen instructions to install CHAOS.

Installing CHAOS on Floppy Disks

If you don't have a hard disk, you can install CHAOS on a dual-floppy-disk computer. Instead of starting CHAOS programs from a single opening menu, you run each CHAOS program directly from the disk.

When you install CHAOS on floppy disks, you can install one or more of the six CHAOS programs on each disk (depending on the size of the disk format you use). The Install program verifies that there is enough room on the disk before it attempts to install the selected program.

To install CHAOS, you'll need several blank, formatted disks.

Follow these steps to use the Install program to install CHAOS on floppy disks:

1. Place disk 1 in drive A and close the drive door.
2. Make drive A the current drive by entering the following:

   ```
   a: ↵
   ```

3. Place a blank, formatted disk in drive B and close the drive door.
4. Start the installation program by entering this command:

   ```
   install ↵
   ```

5. Follow the on-screen instructions to install CHAOS.
6. Label the disk from drive B with the name of the CHAOS program you installed on that disk.
7. Run the Install program multiple times to install all six CHAOS programs.

Increasing the Size of the Environment Space

By default, DOS reserves 128 bytes for the environment space. If insufficient room has been set aside for the environment space, you will see the following message when you boot your computer:

```
Out of environment space
```

If you are running DOS 3.2 or above, you can set an environment variable or specify a SHELL statement in your config.sys file that allocates more space for setting the environment variables. For example, with DOS 3.2 you can use the following statement:

```
SHELL=c:\command.com /e:750 /p
```

This SHELL statement tells DOS which command interpreter to use when booting, and to expand the DOS environment space beyond its 128 byte default to 750 bytes. Refer to your DOS manual for more information about environment space, the SHELL command, and your config.sys file.

Starting CHAOS on a Hard Disk Computer

To start CHAOS: The Software on a hard disk computer, follow these steps:

1. Make sure you're in the directory in which you installed CHAOS. The \chaos directory is the default the install program suggests. If you are currently in another directory on the same drive, enter this at the DOS prompt:

   cd \chaos ↵

   Note: If you used another name for the CHAOS directory, enter that name instead of \chaos.

2. Start CHAOS from the \chaos directory by entering this command:

   chaos ↵

You briefly see a message screen, and then the CHAOS Main menu appears.

The Main menu displays six round *program icons* that represent the six CHAOS programs. There are also six buttons labeled with the names of the CHAOS programs and an Alt-X to Exit button that lets you quit to DOS.

To start one of the CHAOS programs, select its program icon or the button labeled with the program name. For information on how to select buttons and icons, see "Selecting Buttons and Icons" on page 9. You can also press the number next to the program name to start the program. For example, to start The Mandelbrot Sets, press 1.

Note: You can also bypass the CHAOS Main menu to start an individual CHAOS program directly. To do so, make sure you're in the chaos directory, then enter the name of one of the program batch files listed in "Starting CHAOS from a Floppy Disk" on page 9 and press ↵.

Starting CHAOS from a Floppy Disk

To start a CHAOS program from a floppy disk, follow these steps:

1. Place the disk containing the program you want to start in drive A and close the drive door.
2. Make drive A the current drive by entering the following:

   a: ↵

3. Enter the batch filename for the program you want to start and press ↵.

Batch File Name	*Program Name*
mandel	The Mandelbrot Sets
magnets	Magnets and Pendulum
attract	Strange Attractors
game	The Chaos Game
forge	Fractal Forgeries
toy	Toy Universes

Selecting Buttons and Icons

You can choose most CHAOS options by selecting buttons. You can select buttons with the mouse or the keyboard.

If you've installed a mouse with CHAOS, you can work with it exclusively or use it in combination with keyboard commands. For actions such as selecting buttons, you'll find the mouse more convenient to use.

To select program icons and buttons with a mouse, follow these steps:

1. Move the mouse until the on-screen pointer (usually shaped like an arrow) is on the icon or button. The highlighting around the current button changes to indicate that it is selected.
2. Press the left mouse button and release it quickly.

Note: From now on, when you're instructed to select an icon or button, move the pointer to the button or icon and click the left mouse button. This action is called "left-clicking." (Most actions in CHAOS require that you press the left mouse button.)

To select program icons and buttons with the keyboard, follow these steps:

1. Move the pointer or cursor to the icon or button you want to select.

   Depending on where you are in a CHAOS program, you can move around with the following keys:

   - →, ←, ↑, ↓
   - Tab
   - PgUp, PgDn
   - Home, End

   The pointer indicates your current location. As you move the pointer from button to button, their highlighted edges change to indicate which button is currently selected.

2. When the cursor is at the button you want to select, press ↵.

An alternative way to select buttons is to use function keys and keyboard shortcuts. Many buttons in CHAOS programs are labeled with a function key that you can press to select the button. For example, to select the F1 HELP button, you can press F1. Other buttons have a keyboard shortcut that performs the same action as the button. For more information about keyboard shortcuts, see "Using Keyboard Shortcuts" on page 15.

Getting Help in the CHAOS Programs

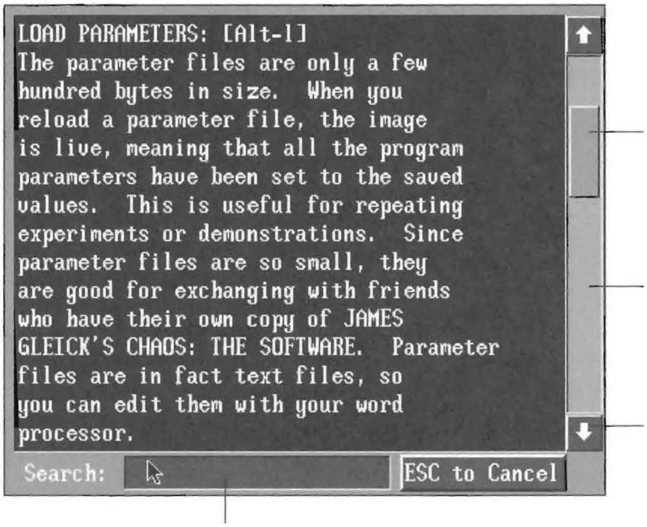
Keyboard shortcut: F1

On-line help is available in all CHAOS programs. CHAOS on-line help includes useful reference material on keyboard commands, program menus, and program terminology.

CHAOS displays help appropriate to your current location in a program. For example, if you're using an Options menu and request help, you get help specific to that menu.

- Select the F1 HELP button or press (F1) to display on-line help.

Help text appears in a window on screen. The following illustration shows a CHAOS help window:



LOAD PARAMETERS: [Alt-1]
The parameter files are only a few hundred bytes in size. When you reload a parameter file, the image is live, meaning that all the program parameters have been set to the saved values. This is useful for repeating experiments or demonstrations. Since parameter files are so small, they are good for exchanging with friends who have their own copy of JAMES GLEICK'S CHAOS: THE SOFTWARE. Parameter files are in fact text files, so you can edit them with your word processor.

Scroll box

Scroll bar

Scroll arrows

Search field

Using Scroll Bars

The scroll box, a rectangular box in the scroll bar, indicates your location in the help text. If the scroll box is at the top of the scroll bar, you're at the beginning of the text. The size of the scroll box is relative to the length of the text. A small scroll box indicates that you currently see a small portion of the help text. If the scroll box is half the length of the scroll bar, then one half of the help text is visible in the window.

You can use the mouse or keyboard to scroll the text in the window.

To scroll with the mouse, do one of the following:

- Left-click on the scroll arrows above or below the scroll bar to move text up or down a line at a time.
- Left-click in the scroll bar above or below the scroll box to move text a page at a time.
- Hold down the left mouse button and drag the scroll box up or down to move text more quickly in the window.

To scroll using the keyboard, do one of the following:

- Press ↓ or ↑ to move text up or down a line at a time.
- Press PgDn or PgUp to move text a page at a time.
- Press End or Home to move to the end or beginning of the help text.

Searching for Help Text

You can use the Search field to locate specific text in the help window. To use the Search field with a mouse, follow these steps:

1. Left-click in the Search field. The Search field changes color and a block cursor appears.
2. Enter the text you want to find.
3. Press ↵. The line where the text appears is highlighted and displayed at the top of the help window.
4. Press / followed by ↵ to locate the next occurrence of the search word.
5. Press ? followed by ↵ to search backwards.

To use the Search field with the keyboard, follow these steps:

1. Press / to search forward or press ? to search backward. The Search field changes color and a block cursor appears.
2. Enter the text you want to find.
3. Press ↵. The line where the text appears is highlighted and displayed at the top of the help window.
4. Press / followed by ↵ to locate the next occurrence of the search word.
5. Press ? followed by ↵ to locate the previous occurrence of the search word.

Exiting the Help Window

To exit the help window

- Right-click anywhere in the help window or press Esc .

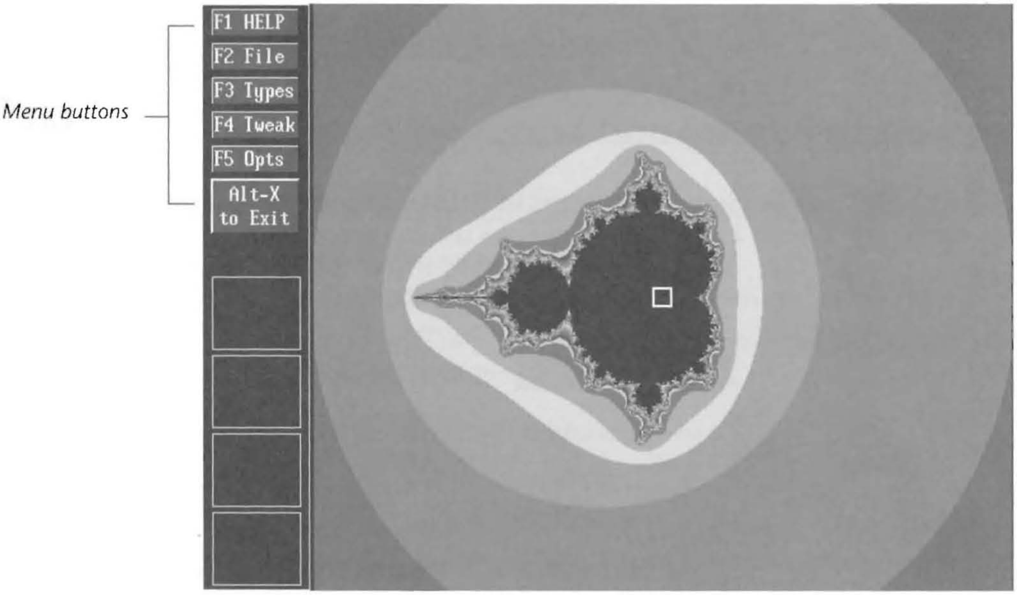
Changing Options and Parameters in CHAOS

You work in the CHAOS programs by changing options and parameter settings to create new images or simulations. You can change settings in CHAOS five ways:

- Using buttons
- Using entry fields
- Using plus (+) or minus (–) buttons
- Using horizontal slider bars
- Using keyboard shortcuts

Using Buttons

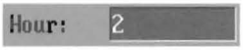
Throughout the CHAOS programs, you choose menu items and change many options by selecting buttons. The opening screen of each program has a set of menu buttons in the upper-left side of the screen. For example, the opening Mandlebrot program screen looks like this:



You can select buttons with either the mouse or the keyboard, as described in "Selecting Buttons and Icons" on page 9.

Using Entry Fields

You can assign specific values to many CHAOS options and parameters by entering information in entry fields. Following is an example of an entry field:



Hour: 2

To use entry fields with the mouse, follow these steps:

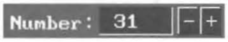
1. Left-click on the entry field. The field turns a different color and a block cursor appears in the field.
2. Press Ctrl+U or use the Backspace key to delete the current entry.
3. Type the new entry and press ↵.

To use entry fields with the keyboard, follow these steps:

1. Move the cursor or highlighting to the entry field.
2. Press ↵. The field turns a different color and a block cursor appears in the field.
3. Press Ctrl+U or use the Backspace key to delete the current entry.
4. Type the new entry and press ↵.

Using Plus (+) and Minus (–) Buttons

Some parameters in CHAOS programs have plus (+) and minus (–) buttons that you can use to increase or decrease the value of the parameter. For example, in Toy Universes, the Number option on the opening screen looks like this:



Number: 31

To use plus and minus buttons with a mouse

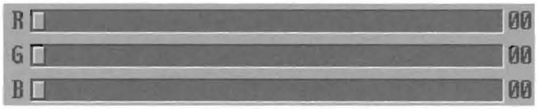
- Left-click the plus (+) or minus (–) button next to the parameter name.

To use plus and minus buttons with the keyboard

- Move the cursor or highlighting to the plus or minus button and press ↵.

Using Horizontal Slider Bars

Some CHAOS parameters have horizontal slider bars for making adjustments. For example, the Palette Editor used in most CHAOS programs includes three slider bars:

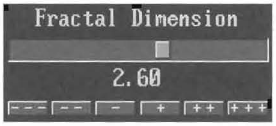


R 00
G 00
B 00

To use slider bars with the mouse, follow these steps:

1. Move the pointer to the slider inside the slider bar.
2. Hold down the left mouse button and drag the slider to the right or left. The parameter value changes to reflect the new position of the slider.

Some slider bars have a set of six decrease/increase buttons below the slider. For example, the Fractal Dimension parameter in the Fractal Forgeries program includes a slider bar that looks like this:



Fractal Dimension
2.60

You can left-click these decrease/increase buttons to move the slider. The greater the number of plus or minus symbols, the larger the move.

To use slider bars with the keyboard, follow these steps:

1. Move the cursor to the slider bar.
2. Press ← or → to move the slider to the left or right.
3. Press Shift + ← or Shift + → to move the slider to the left or right in larger increments.

Using Keyboard Shortcuts

Many options and features in the CHAOS programs have keyboard shortcuts. For example, in The Chaos Game, you can display the next image by selecting a button or pressing N , the keyboard shortcut. Even if you're using a mouse, you'll find it more convenient to use the keyboard shortcut for many features.

Whenever possible, the shortcut is the first letter of the feature or option it represents. For example, in Fractal Forgeries, the shortcut for Animate is (A).

In some cases, the uppercase letter shortcut means the opposite of the lowercase letter shortcut: in The Chaos Game, (N) displays the next image and (Shift)+(N) displays the previous image. In cases where you can increase or decrease a parameter, the lowercase letter shortcut decreases the parameter and the uppercase one increases the parameter. For example, in Magnets, you can decrease the Friction parameter by pressing (F); you can increase the friction by pressing (Shift)+(F).

All six CHAOS programs have common keyboard shortcuts, but each program has additional unique shortcuts. The on-line help and the quick reference card list the shortcuts for each program. You'll also find a keyboard shortcut listed in the manual below the title of the section describing the feature or parameter.

Accepting Changes

Many menus and dialogue boxes in the CHAOS programs have an ACCEPT button. You select the ACCEPT button to confirm your changes and to exit the menu or dialogue box.

Canceling Changes

Many menus and dialogue boxes in the CHAOS programs have a Cancel button. You select the Cancel button to cancel your changes and to exit the menu or dialogue box.

Exiting CHAOS Programs to the Main Menu

Keyboard shortcut: (Alt)+(X)

From any CHAOS program, you can quickly exit to the CHAOS Main menu:

1. Select the Alt-X to Exit button. A message box appears asking you to verify that you want to exit.
2. Select the Yes button or press (Y) to exit.

Quitting to DOS

Keyboard shortcut: Alt+Q

From any CHAOS program, you can also quit the CHAOS program and return to the DOS prompt.

1. Press Alt+Q. A message box appears asking you to verify that you want to exit.
2. Select the Yes button or press Y to exit.

To return to DOS from the CHAOS Main menu

- Select the Alt-X to Exit button.

Chapter 2
The Mandelbrot Sets

> *An eternity would not be enough time to see it all, its disks studded with prickly thorns, its spirals and filaments curling outward and around, bearing bulbous molecules that hang, infinitely variegated, like grapes on God's personal vine. Examined in color through the adjustable window of a computer screen, the Mandelbrot set seems more fractal than fractals, so rich is its complication across scales. A cataloguing of the different images within it or a numerical description of the set's outline would require an infinity of information. But here is a paradox: to send a full description of the set over a transmission line requires just a few dozen characters of code. A terse computer program contains enough information to reproduce the entire set . . . (*Chaos: Making a New Science, *page 221.)*

A not-so-terse program like this one allows one to do far more than reproduce the Mandelbrot set. It lets users become explorers, looking deeper into the heart of this strange object than any mathematician could, barely a decade ago, when it was first discovered. Since then the Mandelbrot set has become an emblem for the new science of chaos, adorning everything from coffee mugs to T-shirts, institutional letterheads to art gallery catalogues.

The computer becomes both a scanner and a microscope. With either a mouse or the keyboard's arrow keys, the user pans across the set looking for the intriguing complexity at the boundary, and then zooms closer and closer in, revealing even more intricate complexity.

The Mandelbrot set is a collection of points (each representing a pair of numbers), shown as black on screen. The colors are like the colors on a contour map of a mountainous region; if one thinks of the set as a plateau, the colors show how steeply the rest of the terrain falls away from it. The program creates its images by testing every point, every pair of numbers. The test is a simple equation repeated over and over again: a pair of numbers goes in, a new pair is computed; the new pair goes in, and the equation produces a new pair; and so on. (The mathematical details are described in "The Mathematics of the Program" on page 42.) It is like feedback through a microphone and speakers—a sound can fade away, or it can grow wildly out of control. If the numbers get larger and larger, running away to infinity, the original point does not belong to the Mandelbrot set. If the original point is in the set, the numbers remain within bounds, either locked in a repeating loop or bouncing chaotically from one place to the next, but never escaping.

Benoit Mandelbrot, the Polish-born mathematician best known as the father of fractal geometry, first discovered this unlikely object in 1979. It was a crude, bulbous shape on his black-and-white computer printouts. Bug-like tendrils stretched away from it. Scattered points lay near the main body, and it was impossible to guess how rich the object really was. It was not even clear whether the whole set—the black region—was a single connected continent or a loose assortment of clustered islands. Mathematicians later proved that it is connected—though one can zoom in forever on the fine filaments connecting one bulb to the next without ever seeing the whole strand.

Every place in the set is unique. Though one gradually begins to sense the shapes and patterns that reappear in different guises, no image ever repeats itself exactly. If one imagines that the full Mandelbrot set is the size of the Earth's Solar System, one can zoom in to see detail the size of individual molecules. Color becomes more and more important, as a signpost showing the way to strands that have not yet come directly into view.

A new kind of geometry has emerged from the exploration of such iterated processes—processes that are simple in themselves, but create complexity through repetition and feedback. Computers are essential, with their ability to repeat computations at high speed and display the results graphically. Traditional forms of geometry take equations and ask what numbers satisfy them—the sets that result form the circles, ellipses, and parabola familiar from high-school geometry. But when a geometer iterates an equation instead of solving it, the equation becomes a process instead of a description. There is good reason to believe that nature operates this way, with simple processes repeated again and again, on large scales and small.

The sheer intricacy of the Mandelbrot set, abstract though it is, has already helped change scientists' preconceptions about the world. The lesson of the Mandelbrot set is a lesson about boundaries, especially the sorts of boundaries that arise in complicated processes. An engineer thinking about the design of a bridge must worry about a boundary that exists between safety and catastrophe. That boundary depends on such things as the speed of the wind and the weight of the load that the bridge must carry. An electric power company has to worry about a boundary that exists between stability and instability in a network of cables and generating stations spanning thousands of miles. Or a person trying to make a decision between two important, attractive, but incompatible alternatives might imagine a sort of boundary between them.

And what sort of boundary will they imagine? The point is—and it came as a surprise—that such boundaries are not necessarily smooth. They may be extremely complicated, so that some slight change in circumstances throws one into an altogether different region. A bolt of lightning strikes, or a rat gnaws through a cable, and suddenly 20 million households are wondering why the lights are out. Indeed, boundaries much like the rim of the Mandelbrot set appear in a variety of real physical problems.

Julia Sets

> *Some are like circles that have been pinched and deformed in many places to give them a fractal structure. Others are broken into regions, and still others are disconnected dusts. But neither words nor the concepts of Euclidean geometry serve to describe them. The French mathematician Adrien Douady said: "You obtain an incredible variety of Julia sets: some are a fatty cloud, others are a skinny bush of brambles, some look like the sparks which float in the air after a firework has gone off . . ."* (Chaos: Making a New Science, *page 221.)*

A simpler, older cousin of the Mandelbrot set is the group of objects called Julia sets, after Gaston Julia, an early twentieth-century French mathematician who conceived them long before computers made it possible to see their variety. Julia sets are more symmetrical and more straightforwardly fractal—that is, tending to repeat their structure on smaller and smaller scales.

Each Julia set is associated with a single complex number and therefore with a single point on screen, its starting point. The Mandelbrot set itself serves as a sort of guide to Julia sets, a sort of catalogue. The most interesting Julia sets are those associated with points near the rim of the Mandelbrot set. Points in the Mandelbrot set produce Julia sets that are connected, though sometimes with great intricacy, like the Mandelbrot set itself. Points outside the Mandelbrot set, however, produce Julia sets that are disconnected. They are fractal "dusts," clusters of shapes and specks with their own special brand of self-similarity across scales, like stars and galaxies scattered and clustered through the universe.

Cubic Mandelbrots, Cubic Julias, and the Rudy Set

A few years ago, drawing a Mandelbrot set pushed computers to their limits. With new, faster machines widely available, generating an ordinary Mandelbrot set has become less challenging; one can in fact "fly around" a Mandelbrot set in real time on some of the larger computers. In order to keep the game interesting, mathematicians have proposed a simple change in the formula used to define the Mandelbrot set: replace an exponent of two by an exponent of three. The new formula leads to the cubic Mandelbrot sets and the cubic Julia sets.

There is only one standard Mandelbrot set, but there are infinitely many cubic Mandelbrot sets. By changing two control parameters, one can scan through a "stack" of these sets and imagine piling them up to produce a three-dimensional (or even a four-dimensional!) fractal object.

The cubic Mandelbrot sets have their own special symmetry, the symmetry of a letter *S* rather than the symmetry of a letter *D*. Each of them has the full richness of the standard Mandelbrot set; remarkably enough, many of them contain small copies of the standard Mandelbrot. By varying one of the control parameters, one can occasionally make the small Mandelbrots split open like ripe seedpods.

Associated with each point of a cubic Mandelbrot set is a cubic Julia set. These objects can be wholly asymmetrical, and some of them break into several separate connected pieces. The visual investigation of these objects has only just begun, and many new shapes are waiting to be found.

Mathematicians with computers are a bit like explorers with ships, and like explorers they enjoy naming new features of their esoteric terrain. In the process of writing the algorithms for computing the cubic Mandelbrot sets, Rudy Rucker saw a way of combining information about all of these sets into a single set, which he then dubbed the Rudy set. This set exists in the parameter space of the cubic Mandelbrot sets; choosing cubic Mandelbrot settings from inside the Rudy set leads to cubic Mandelbrot sets that are non-empty.

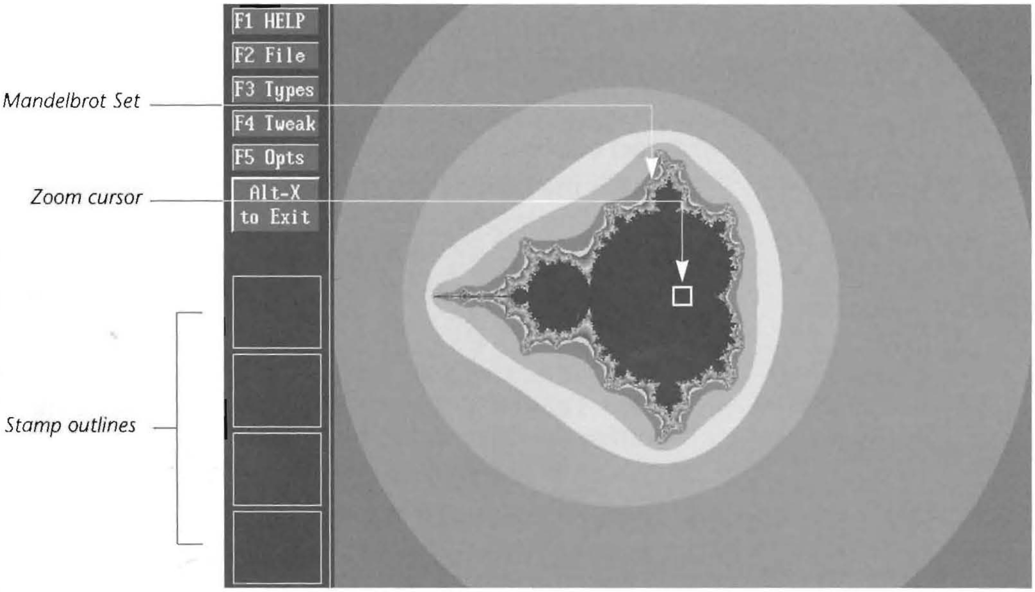
Things To Try

Zoom in again and again on the interesting regions of the boundary (use the left mouse button or press Ins). Change the color palette (press Alt+A) or the width of the color bands (press W) to reveal new features of the landscape. The small windows at the left show where you've been, and left-clicking one of them returns you to that image.

You can explore a new fractal type by pressing E for a Cubic Mandelbrot set or R for the Rudy set. Pressing M returns you to the Mandelbrot set. From any place in the Mandelbrot set, you can see the corresponding Julia set by pressing J.

Starting the Program

When you start The Mandelbrot Sets, you see the following screen:



The main image area of the screen shows the *Mandelbrot set,* a complex, two-dimensional object showing equations repeated again and again. The small white box in the middle of the Mandelbrot set is the *zoom cursor.*

On the lower left, you see four rectangular outlines called *stamps.* The program stores stamp-sized versions of your four most recent images in the stamp outlines.

Note: The program paints the screen four times (coarse to fine) to paint all the pixels in a fractal image. You can make changes to an image any time while the program is painting it.

Exploring the Program

You can have fun exploring the five kinds of sets mentioned at the start of this chapter. Try out several features without worrying about understanding each feature thoroughly. You learn how to use these features later in this chapter.

Suggestion: Use the parameter status line when you first work with the program, to help you identify different types of sets. To display the parameter status line, press Alt+I twice (until you see the name of the current set on the left side of the status line). For details on the status lines in this program, see "Status Line" on page 41.

First, display other sets in the program.

1. Start by pressing X to change from a zoom cursor to a select cursor.
2. With the Mandelbrot set displayed at the start of the program, press J to display a Julia set.

   Notice the stamp image that appears in the lower-left stamp outline. For details on stamps, see "Using Stamps in the Program" on page 26.
3. The Julia set you see depends on the select cursor's position. Try moving the select cursor and left-clicking to see a different Julia set.
4. Press E to display a cubic Mandelbrot set.
5. Press R to display the Rudy set.
6. Press M to display the Mandelbrot set again.
7. Return to the first cursor in the program, the zoom cursor, by pressing X again.

Next, try some of the color options in The Mandelbrot Sets.

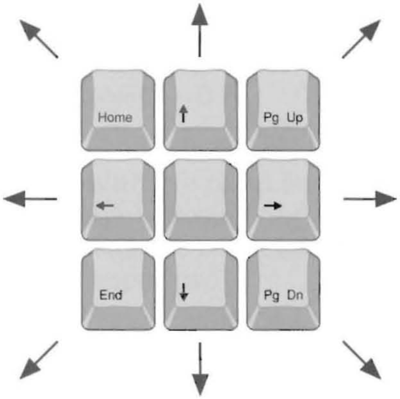
1. With the Mandelbrot set displayed, press N to change the fill type to Bullseye fill. Press N again to change to Feathers fill. Press N again to return to Blank fill.
2. Press Alt+J to change the image's color. Cycle through additional color options for the image by continuing to press Alt+J.
3. Press Alt+D to return to the default color palette.

You can experiment with sound, too.

1. Press E to display a cubic Mandelbrot set.
2. When the program begins to paint the set, press Alt+V to turn sound on, and watch all four screen-paints. You hear the tone change as the scan lines move to new colors. The tone becomes complex-sounding as the lines move over multicolored areas.
3. Left-click or press Ins to zoom in once, and listen to the tone as the program paints the magnified image.
4. Press Alt+V to turn the sound off again.

Moving Around in the Image Area

You can use the mouse or the keyboard to move the cursor around in the image area. To move around using the keyboard, make sure Num Lock is turned off and refer to the following illustration:



Numeric keypad keys and direction of cursor movement

Using Stamps in the Program

The Mandelbrot Sets program generates stamps for changes you make, such as zooming in or selecting a new type of set. (The program does not generate stamps when you make changes to the palette.) You can select one of the four images stored in the stamps to display in the image area.

1. Move the cursor to the stamp you want.
2. Left-click or press ↵ to display the new image.

   You can also press Alt+1, Alt+2, Alt+3, or Alt+4 to select a stamp image to display in the image area (with stamp 1 being at the bottom and stamp 4 at the top).

You can also get parameter information on any of the four stamp images.

1. Right-click on a stamp icon to view the parameter information.

   If you're not using a mouse, you can press Alt+5, Alt+6, Alt+7, or Alt+8 to display a stamp information box (with stamp 5 being at the bottom and stamp 8 at the top).
2. When you've read the parameter information, click either mouse button or press any key to remove the stamp information box from the screen.

Zooming, Panning, and Centering

Zooming In on an Image

You can zoom in on areas that interest you. For example, you might want to examine self-similar areas in greater detail.

1. Move the zoom cursor (the small white box) to the location you want. (The cursor location becomes the center of the magnified image.)
2. Left-click or press Ins.
3. To zoom out, right-click or press Del.

Note: Normally, The Mandelbrot Sets program runs in fast mode. When you zoom down to small sizes, the program must switch to a slower mode. A low-pitched beep signals when the program switches to the slower mode, and a high-pitched beep signals when the program switches back to the fast mode.

In addition to using the regular zoom cursor, you can use the *zoom box*. You can resize the zoom box to enclose any area you want to magnify.

1. Press Alt+Z or the spacebar to display the zoom box cursor.
2. Move the zoom box to the location you want to zoom in on.
3. Hold down the Shift key and move the mouse to resize the zoom box.

   You can also resize the zoom box using the keyboard. Make the zoom box larger by pressing Shift+↑ or Shift+←. Make it smaller by pressing Shift+↓ or Shift+→.
4. Left-click or press Ins to zoom in on the specific area.
5. With the zoom box displayed, you can also zoom out by right-clicking or pressing Del. (After you zoom in or out with the zoom box, the cursor you were previously using reappears.)
6. If you change your mind when working with the zoom box, you can press Esc to return to the cursor you were previously using.

Zoom Box Status Line

When you turn on the zoom box and move the cursor, a status line automatically appears. You can use the zoom box status line to calibrate the exact size of a window for zooming. The first two numbers in the status line show the X and Y coordinates that mark the center of the zoom box; these numbers change constantly as the cursor moves. The third number shows the width of the window you are defining with the zoom box. The window-width number changes as you enlarge or shrink the zoom box to define the window.

Panning an Image

When you pan an image, you can see the parts of it that are outside the edges of the image area. For example, you might want to pan after you zoom in to a "fringe" part of an image.

1. Move the cursor to the location you want to pan to. (The cursor location becomes the center of the panned image.)
2. Press Alt+O to pan to the new location.

Centering an Image

As you work with images, you might feel unsure of where you are in the set. Centering an image moves it to the center of the image area, unzooms it, resets any parameters you've changed, and resets the fractal type to either Mandelbrot or cubic Mandelbrot.

- Press C.

You can also center an image without resetting parameters.

- Press Alt+C.

Displaying Sets with the Select Cursor

You can change from the zoom cursor to the *select cursor* (a white box with a diamond inside) and choose other types of sets to display.

Note: You cannot display the Rudy set by using the select cursor.

To use the select cursor to display other sets, first do the following:

1. Press (X) to display the select cursor.
2. Move the cursor to the area whose parameters you want to use to calculate the new set.

The following table describes how to work with the select cursor.

If You Want To Display . . .	And You Currently Are Displaying . . .	Do This
Mandelbrot	Julia	Right-click or press (Del)
Julia	Julia or Mandelbrot	Left-click or press (Ins)
Cubic Mandelbrot	Cubic Mandelbrot, Rudy, or Cubic Julia	Right-click or press (Del)
Cubic Julia	Cubic Julia, Rudy, or Cubic Mandelbrot	Left-click or press (Ins)

Creating New Images

The Mandelbrot Sets program lets you create variations of the five types of sets in it. For the Mandelbrot and Rudy sets, you can alter only the fill type. For the other sets, you can tweak additional parameters that change the shapes of the sets.

Viewing Preset Images

The program comes with 21 preset images that are variations on the basic sets. These preset images give you a good idea of some of the images you can create.

1. Press (Alt)+(N) to view the first preset.

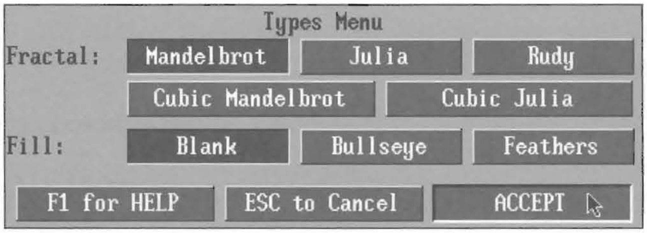
2. Continue to press Alt+N to view all 21 presets.

Note: You can also press + to cycle forward through presets one at a time, or – to cycle backward.

Selecting Fractal and Fill Types

To create a new image, begin by selecting the *fractal type* and *fill type* you want. A fractal type is an object such as a Julia set, the Rudy set, and so on. A fill type is a pattern of colors (or no colors) that you add to the interiors of Mandelbrot, Rudy, and cubic Mandelbrot sets only.

To select fractal and fill types, left-click the F3 Types button at the upper-left corner of the screen, or press F3. The Types menu appears:



Selecting a Fractal Type

The Mandelbrot Sets program has five different fractal types for you to explore. To display the fractal type you want, select one of the following buttons on the Types menu:

- Mandelbrot
- Julia
- Rudy
- Cubic Mandelbrot
- Cubic Julia

You can also select a fractal type with keyboard shortcuts (shown in the sections that follow), or with the select cursor. For details on using the select cursor, see "Displaying Sets with the Select Cursor" on page 29.

Mandelbrot

Keyboard shortcut: (M) (with any set displayed)

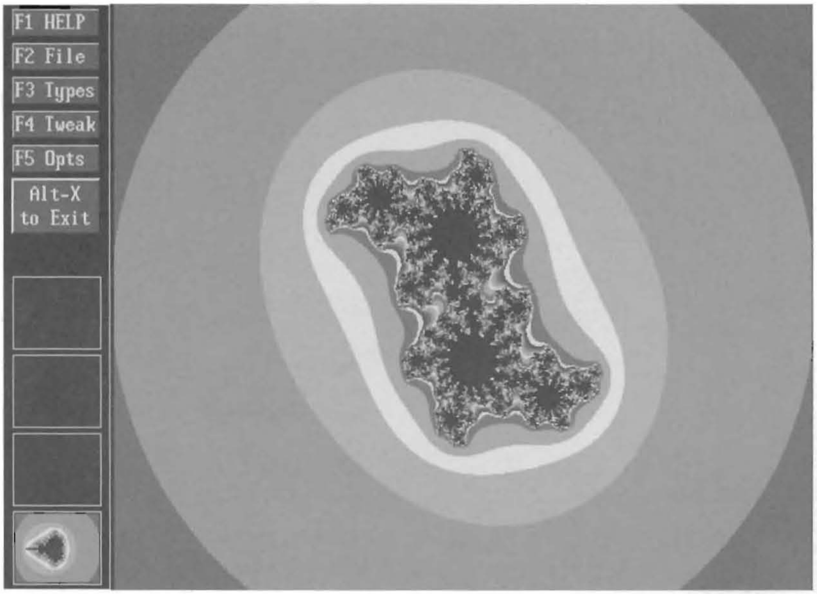
The Mandelbrot set is made up of a collection of points (the black middle area), each of which represents a pair of numbers. The program creates its image of the set by testing each pair of numbers repeatedly.

Julia

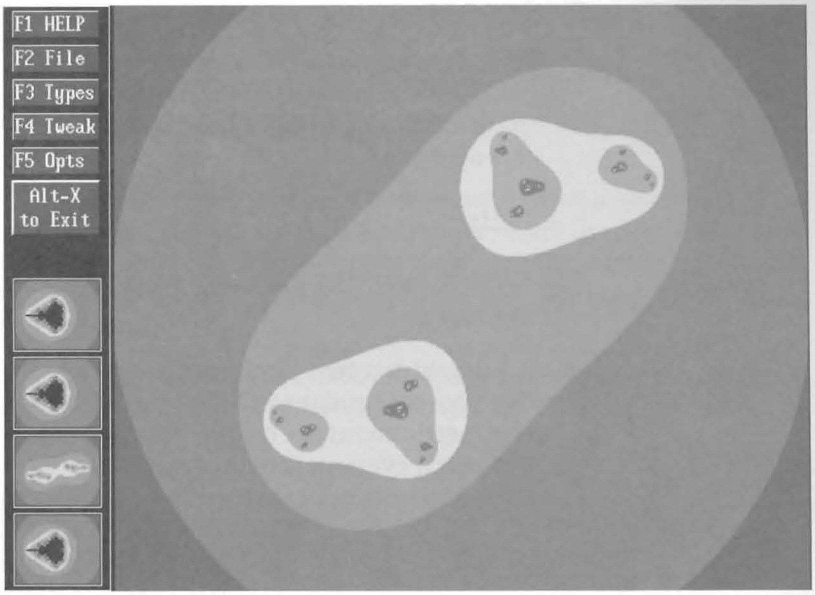
Keyboard shortcut: (J) (only with a Julia set or the Mandelbrot set displayed)

A Julia set is a two-dimensional fractal object formed by an iterative process similar to that of the Mandelbrot set. All Julia sets are symmetric about the origin, like the letter *s*. In *connected* Julias, you can reach every part from every other part. In *disconnected* Julias, there is no all-black path from any point to any other point.

Here's an example of a connected Julia set:



And here's an example of a disconnected Julia set:



Suggestion: Explore connected and disconnected Julia sets. With the select cursor displayed, move the cursor into the black interior of the Mandelbrot set. Left-click or press (Ins) to see a connected Julia set. Right-click or press (Del) to return to the Mandelbrot set. Then move the cursor to an area far outside the Mandelbrot set. Left-click or press (Ins) to see a disconnected Julia set.

Rudy

Keyboard shortcut: (R) (with any set displayed)

The Rudy set tries to catalogue all the cubic Mandelbrot slices.

Suggestion: The details of the Rudy set are quite complicated. With the zoom cursor displayed, try zooming in on any of the set's edges. Then, with the select cursor displayed, move the cursor into the black interior of the set. Right-click or press (Del) to see a nonempty cubic Mandelbrot. Press (R) to display a Rudy set again, and repeat the experiment with a different cursor location in the black interior.

Cubic Mandelbrot

Keyboard shortcut: E (with any set displayed)

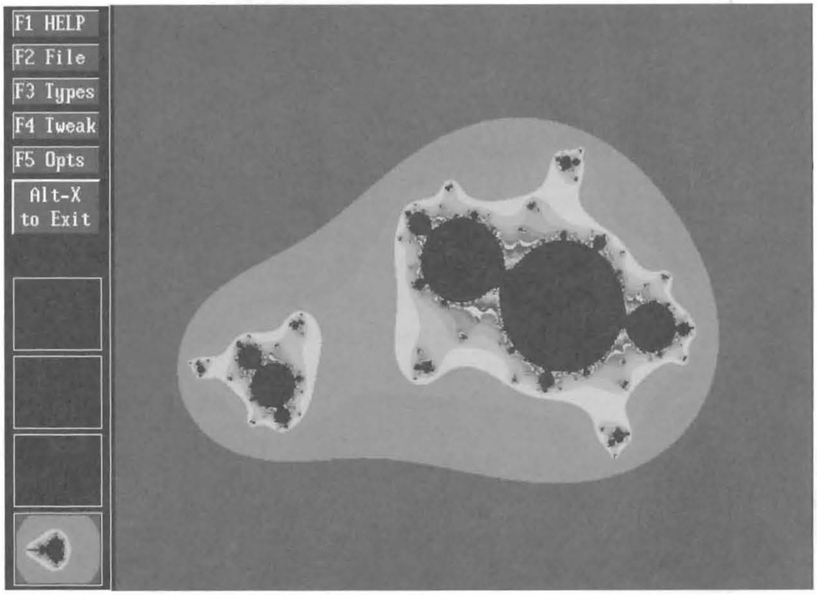
A cubic Mandelbrot is made up of a collection of points, each of which represents a quadruple of numbers—the *A, B, U,* and *V parameters*. Two of the parameters are the same for each point, and two vary from pixel to pixel. The program creates its images by repeatedly testing each quadruple of numbers.

Cubic Julia

Keyboard shortcut: J (only with the Rudy set or a cubic Mandelbrot set displayed)

Like Julia sets, cubic Julias can be either connected or totally disconnected. There is also a third possibility: cubic Julias can be *partly disconnected,* meaning that they break up into distinct solid chunks. Unlike Julia sets, cubic Julias can be completely asymmetric.

Here's an example of a partly disconnected cubic Julia set:



Selecting a Fill Type

CHAOS lets you choose from three different fill types—Blank, Bullseye, and Feathers—for the interior of the Mandelbrot, Rudy, and cubic Mandelbrot sets. The *Blank* fill type (no pattern of colors) is the default in the program. The *Bullseye* fill type displays a pattern of concentric colors, like contour lines on a map. The *Feathers* fill type is a pattern of plume-like or fan-like colors.

To display the fill type you want, select one of the following buttons on the Types menu:

- Blank
- Bullseye
- Feather

You can also use a keyboard shortcut to cycle through fills for the Mandelbrot set.

- Press N.

You can cycle through fills for the Rudy set.

- Press S.

And you can cycle through fills for cubic Mandelbrot sets.

- Press F.

Note: For more details on the calculations for fill types, see "The Mathematics of the Program" on page 42.

Tweaking Parameters

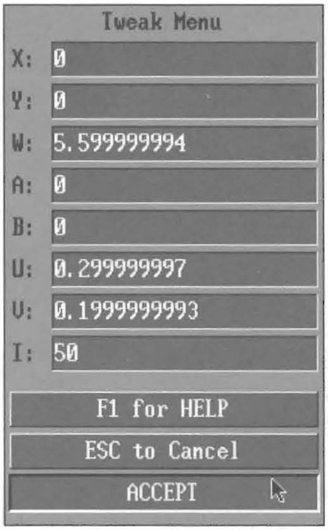
You can use the Tweak menu to adjust an image the way you want. You can also enter published parameters using the Tweak menu. For example, you can enter the following parameters to see the region known as Peitgen's Shepherd Crook in the Seahorse Valley of the Mandelbrot set: X = **–0.7452**; Y = **0.1126**; W = **0.008**; I = **300**.

CHAOS lets you tweak these parameters:

- X, Y, and W (for position and width of the window through which you view sets)
- A, B, U, and V (for shapes of cubic Mandelbrots, Julias, and cubic Julias)
- I (for iteration count controlling the speed and accuracy of calculations)

Note: For the W parameter, and for the A, B, U, and V parameters, you can enter a setting in exponential notation, such as **7.8e–06**. If you enter a setting in decimal figures, such as **.00004**, the program converts it to exponential notation (in this case, **4e–05**).

To tweak parameters, left-click the F4 Tweak button at the upper-left corner of the screen, or press F4. The Tweak menu appears:



Tweak Menu

X: 0

Y: 0

W: 5.599999994

A: 0

B: 0

U: 0.299999997

V: 0.1999999993

I: 50

Changing the Window Position

Tweaking the X and Y parameters sets the position for the center of the window through which you view all sets. The default setting for both X and Y parameters is 0, which centers the image on screen. You can enter X and Y parameter settings between –2 and 2.

- Enter the new value for X, Y, or both. Then press ↵.

Changing the Window Width

Tweaking the W parameter sets the width of the window through which you view all sets. For example, making the window 1/2 as big has the effect of zooming in. The default setting for the W parameter is 5.6.

You can enter a W parameter setting between 1e–60 and 5.6. The lowest limit for the W parameter, 1e–60, is equal to $1 * 10^{-60}$ (in decimal format, this number would be written as a decimal point followed by 59 zeros followed by a 1).

Note: If you set this parameter to a low value (below 1, for example), you might see only the black interior of a set.

- Enter the new value for W. Then press ↵.

Changing the Shape of Cubic Mandelbrot Sets

Keyboard shortcut: Shift+A and Shift+B to increase; A and B to decrease

You can think of a cubic Mandelbrot as a slice of a four-dimensional object. If you change A, B, or both, you change where the program makes the slice. This, in turn, affects the shape of the set you view.

- Enter the new value for A, B, or both. Then press ↵.

Note: If you've selected a different slice type from the default UV slice, then you use the keys *not* in the name of the slice type to change the shape of the cubic Mandelbrot. For example, if you are working with the 4D Slice option set to BU, then you use the A and V keys to change the shape of the cubic Mandelbrot. For details on working with 4D slice, see "4D Slice" on page 40.

Changing the Shape of Julia Sets

Keyboard shortcut: Shift+U and Shift+V to increase; U and V to decrease

Changing U, V, or both parameters chooses different iteration starting-points within the Mandelbrot set for a Julia set.

- Enter the new value for U, V, or both. Then press ↵.

Changing the Shape of Cubic Julia Sets

Keyboard shortcut: A, B, U, and V to decrease; Shift+A, Shift+B, Shift+U, and Shift+V to increase

When you tweak A, B, U, and V parameters for a cubic Julia set, you select different constants in the four-dimensional space that contains the full four-dimensional object.

- Enter the new value for A, B, U, V, or any combination. Then press ↵.

Suggestion: There are so many different cubic Julias that you might have trouble finding some of them again. Use the Save Parameters button on the File menu to save your most interesting parameters. For details on saving files, see "Saving and Loading Files" on page 209.

Changing the Iteration Count

Keyboard shortcut: Shift + I to increase; I to decrease

The I parameter changes the iteration count, which is the maximum number of steps per pixel that the program uses in its calculations for sets. When you set high values for I, the program calculates sets slowly but with higher precision. Setting a high iteration count also generally reduces the amount of black on screen. For example, some black interiors of sets might turn into spirals of colors. When you set low values for I, the program calculates sets quickly but with less precision. The default settings are 50 for the Mandelbrot and Julia sets, and 20 for the Rudy, cubic Mandelbrot, and cubic Julia sets. You can enter an I parameter setting between 10 and 30000.

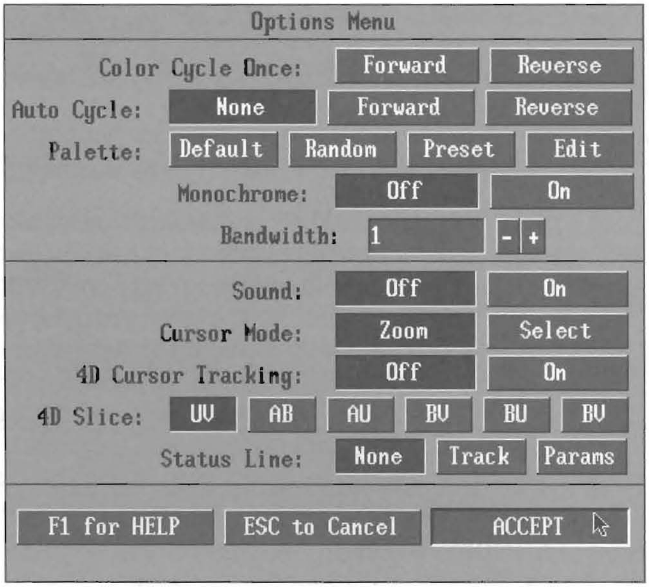
Note: Whenever you zoom in or out, the program estimates a reasonable value for the I parameter. If you change the W parameter, which sets window width, the program does not automatically adjust the I parameter to match it. A good rule of thumb in this case is to change the I parameter to about 100 times the number of decimal places out to the first nonzero digit in the W parameter you set. For example, if you enter **.002** for the W parameter, you count three places to the right of the decimal, multiply 3 * 100, and enter **300** for the new I parameter.

- Enter the new value for I. Then press ↵.

Using the Options Menu

Use the Options menu to turn on color, sound, and status line, and to adjust bandwidth, change cursor mode, and make a slice through various angles of 4D space.

To select options, left-click the F5 Opts button at the upper-left corner of the screen, or press F5. The Options menu appears:


Options Menu
Color Cycle Once: Forward Reverse
Auto Cycle: None Forward Reverse
Palette: Default Random Preset Edit
Monochrome: Off On
Bandwidth: 1 - +
Sound: Off On
Cursor Mode: Zoom Select
4D Cursor Tracking: Off On
4D Slice: UV AB AV BV BV BV
Status Line: None Track Params
F1 for HELP
ESC to Cancel
ACCEPT

Color Options

The following color options are common to five of the CHAOS programs. For information on these options, see "Using Color Options" on page 214.

- Color Cycle Once
- Auto Cycle
- Palette (Default, Random, Preset, Edit)
- Monochrome

Bandwidth

Keyboard shortcut: Shift + W to increase; W to decrease

Changing bandwidth affects the lively, flame-like areas near the interior of the sets. Increasing bandwidth pushes the flame-like areas out from the interior; decreasing bandwidth pulls them in toward the interior. The default setting is 1. You can set bandwidth between **0.001** and **32767**.

- Enter the new value for bandwidth and press ↵. You can also raise and lower bandwidth by using the + and − buttons next to the Bandwidth option.

Sound

Keyboard shortcut: Alt+V

When you turn sound on in The Mandelbrot Sets, you hear a tone as the pixel-scanning lines move down through the set in the image area. This tone changes as the scan lines move to new colors. Sound becomes especially complex as the scan lines move over *boundary,* or multicolored, areas. When the scan lines are moving through a black region, the tone becomes a low buzz. On the fourth and final scan of the image, the rising and falling of the tone is smoother.

- Select the On or Off button to toggle sound in the program.

Suggestion: Listen to the sound that the program produces when you change the fill type for the Mandelbrot set. With the Mandelbrot set displayed, press Alt+V to turn on sound. Immediately press N to change to Bullseye fill and listen to the sound as the program scans the image. After the final scan, press N again to change to Feathers fill and listen to the sound for this image.

Cursor Mode

Keyboard shortcut: X

As you work in this program, you can use the zoom cursor or the select cursor. The zoom cursor is a small white box; the select cursor is a white box with a diamond inside. To find out about working with the zoom cursor, see "Zooming In on an Image" on page 27. For details on working with the select cursor, see "Displaying Sets with the Select Cursor" on page 29. The default setting is zoom cursor.

- Select the Zoom or Select button to work with the cursor you want.

4D Cursor Tracking

Keyboard shortcut: D

Turning on 4D cursor tracking when you are working with the 4D Slice option lets you keep a particular cubic Julia set value at the center of the screen as you switch from one plane to another. Remember to turn this option off after you finish working with 4D Slice, or any zoom you make will not work correctly. The default setting is 4D cursor tracking turned off.

- Select the On or Off button to toggle 4D cursor tracking.

Suggestion: With 4D cursor tracking turned on, work with the select cursor to display different cubic Julia sets. First, left-click or press Ins to set a cubic Julia set's parameters. Press L to change to a different slice type. (See the following section for a discussion of slicing along different planes with the 4D Slice option.) Then left-click or press Ins to set different cubic Julia parameters. Continue to alternate between pressing L to change slice types and left-clicking or pressing Ins to set different cubic Julia parameters.

4D Slice

Keyboard shortcut: L to move backward through list of planes; Shift + L to move forward through list of planes

The cubic Mandelbrot sets are all slices of a single 4D object. In mathematics, this object is called the Cubic Connectedness Map, or CCM. The CCM is the set of all parameter quadruples (A, B, U, V) such that the cubic Julia set determined by these parameters is connected.

When you use the 4D Slice option, you can slice the CCM along the following list of planes: UV, AB, AU, BV, BU, and AV. The default setting is the UV plane. For details on the CCM and the calculations involved in slicing along different planes, see "The Mathematics of the Program" on page 42.

- Select the UV, AB, AU, BV, BU, or AV button to slice along the plane you want.

Suggestion: You might want to investigate the different slices of cubic Mandelbrots by using the select cursor. Right-click or press Del to try different parameter combinations for a given slice type.

Status Line

Keyboard shortcut: Alt + I

The Mandelbrot Sets program has three different status lines—track, parameters, and zoom box. You can use the Status Line option to turn the track and parameters status lines on and off. You might want to work with the status lines off so you can see the beauty of the Mandelbrot sets clearly, with no clutter at the bottom of the screen. The default setting is status line turned off.

Note: For more information on the zoom box status line, see "Zoom Box Status Line" on page 28.

To display the status line you want, select one of the following buttons on the Options menu:

- None
- Track
- Params

The program displays a different *track status line* depending on the cursor mode you're working with. The track status line for the zoom cursor shows the X and Y coordinates of the cursor in parentheses. These numbers change constantly as the cursor moves. The numbers in square brackets show the width that the window will be when you zoom in or out. When you turn on this status line for the zoom cursor, you can use it to calibrate exactly where you are working as you zoom in and out. This might be useful, for example, when you try to re-create different Julia sets, or when you try to study exactly the same spots in the Mandelbrot set.

When you work with the select cursor, the track status line shows how the A, B, U, and V parameter settings change relative to the current 4D Slice option. The parameter settings appear beside uppercase A=, B=, U=, and V= in the status line. For example, if you set 4D Slice to AU and display a cubic Mandelbrot set, you can use this status line to watch how different settings of B and V create different-looking cubic Mandelbrots.

The *parameter status line* shows the current fractal type, followed in order by these parameter settings from the Tweak menu:

- X and Y window-position parameters in parentheses
- W window-width parameter
- A, B, U, and V parameters beside lowercase a=, b=, u=, and v=

This status line helps you identify how different A, B, U, and V parameter settings affect the set. You can also use the status line to watch the effects of changing X, Y, and W settings. In addition, if you display this status line when you save an entire screen as a *.gif* file, you can return to the same settings that you see in the parameter status line when you reload the *.gif* file.

Saving and Loading Files

Use the options on the File menu to save and reload parameter files and *.gif* files. For information on saving and reloading files, see "Saving and Loading Files" on page 209.

The Mathematics of the Program

Computing Sections of the Cubic Connectedness Map

A map in the plane is some system for finding an image P' of each point P. You might, for instance, stretch the plane by moving each P twice as far from the origin O, or you might rotate the plane, moving each point P to a point P' some fixed number of degrees further clockwise.

If you think of each point in the plane as a complex number $z = x + yi$, you can get a very interesting map by taking z into z^2. If you keep in mind that "i" is meant to be an "imaginary" number such that $i^2 = -1$, then a little algebra reveals that $z^2 = x^2 - y^2 + (2xy)i$. In geometric terms, the operation of letting P' be the square of P works by actually squaring the length from P to the origin and by then rotating P to P' in such a way that the angle between OP' and the x-axis is exactly twice the angle between OP and the x-axis.

If f is a map in the plane, and f maps z into z', we can express this either by writing $z' = f(z)$ or by writing $z \xrightarrow{f} z'$. Given an f and a z, you can define a sequence z_n by:

$z_0 = z$

$z_{n+1} = f(z_n)$.

In terms of f:

$z \xrightarrow{f} z_1 \xrightarrow{f} z_2 \xrightarrow{f} z_3 \xrightarrow{f} \ldots$

The norm $|z|$ of $z = x + iy$ is obtained by $\sqrt{x^2+y^2}$ and is equal to the distance of z from the origin. If $|z_n|$ becomes arbitrarily large as n increases, you say that z goes to infinity under the map *f*, which you can symbolize as $z \xrightarrow{f} \infty$. If there is some real BOUND so that for all n, $|z_n|<$ BOUND, then you say that z is bounded under the map *f*, and you write $z \xrightarrow{f}$FINITE. Note that $z \xrightarrow{f}$FINITE does not mean that the z_n sequence approaches a specific limit; more characteristically, the z_n will cycle among being close to some finite number of different limits.

The Julia set for a map *f* is defined as the set of all *z* in the plane that are bounded under *f*. Symbolically, the Julia set for *f* is $\{z : z \xrightarrow{f}$FINITE$\}$.

The map *fc* given by $fc(z) = z^2 + c$ has been widely studied. The *fc* maps are in some sense the "only" quadratic maps because by changing your coordinate system, any general quadratic map $Aw^2 + Bw + C$ can be put into the form $z^2 + c$, with *z* a linear function of *w*. The Julia set for *fc* is named Jc.

Jc = $\{z : z \xrightarrow{fc}$FINITE$\}$.

All the Jc are symmetric about the origin, meaning that if the complex number $z = x + iy$ is in Jc, then the complex number $-z = -x - iy$ is in Jc as well. Some of them are connected with nonempty interiors (disklike), some are connected with empty interiors (linelike), and the others are totally disconnected (dustlike). Jc being totally disconnected means that if A and B are in Jc, there is a disk around A that hits no member of Jc and that excludes B. "You can't get there from here." No quadratic Julia set Jc is ever broken into just a few pieces. If it is disconnected, it is totally disconnected. Given this fact and the fact that the Jc have symmetry around the origin, it's not too surprising to learn that Jc is connected if and only if the origin (0,0) is a member of Jc.

If you imagine varying c through all possible complex values u+vi, you can think of a c-plane that has a Jc set defined at each of its points. Benoit Mandelbrot had the idea of looking at the locus of all the c's for which Jc is connected. This is the Mandelbrot set, M. Symbolically,

M = { c : Jc is connected};
  = { c : the origin is in Jc }, because Jc is connected ↔ (0,0) in Jc;
  = { c : $0 + 0i \xrightarrow{fc}$FINITE}, by the definition of Jc.

The maps that the cubic Julias and cubic Mandelbrots are based on have the form *f*kc, with $fkc(z) = z^3 - 3kz + c$. By translating the origin, any cubic map $Aw^3 + Bw^2 + Cw + D$ can be put into the form $z^3 - 3kz + c$, with *z* a linear function of *w*. (The reason for casting the linear coefficient in the form -3k is that this k is used to calculate cubic Mandelbrots.) For each fkc you can define a cubic Julia set Jkc by:

$Jkc = \{z : z \xrightarrow{fkc} \text{FINITE}\}.$

Unlike in the quadratic case, these Julia sets are generally not symmetric. Some of them are connected, some are disconnected but not totally disconnected (made of numerous separate connected patches), and some are totally disconnected. It has been proved that Jkc is connected if and only if both the complex numbers $k = a + ib$ and $-k = -a - ib$ are in Jkc. As Jkc is not symmetric, it may happen that only one of k or -k is in Jkc. Jkc is connected only when both of these critical points are in Jkc.

The four-dimensional set of all complex pairs k and c such that Jkc is connected is known as the Cubic Connectedness map, or the CCM. The CCM is four-dimensional since it lies in a space with four parameters: a,b,u,v.

$CCM = \{(k,c) : Jkc \text{ is connected}\};$
$\quad = \{(k,c) : (k \xrightarrow{fkc} \text{FINITE}) \text{ AND } (-k \xrightarrow{fkc} \text{FINITE})\}$

The way the program depicts the CCM is to show various two-dimensional cross-sections of it. These cross-sections are called cubic Mandelbrot sets. If, for instance, $k = a + bi$ is fixed, you speak of the cubic Mandelbrot set Mk. Mk is a uv-plane slice of the CCM, and

$Mk = \{c : Jkc \text{ is connected}\};$
$\quad = \{c : (k \xrightarrow{fkc} \text{FINITE}) \text{ AND } (-k \xrightarrow{fkc} \text{FINITE})\}$

Note that computing an Mk is harder than computing the standard quadratic Mandelbrot set M in two ways. First of all, the map *f*kc involves cubing complex numbers, where the map *f*c only squares the complex numbers it deals with. Secondly, to check if c is in M, you need only check for one condition: $0 + 0i \xrightarrow{f} \text{FINITE}$; but to check if c is in Mk you need to check two conditions: $k \xrightarrow{fkc} \text{FINITE}$ and $-k \xrightarrow{fkc} \text{FINITE}$. On the other hand, if a cubic *f*kc grows, it grows explosively, so you can make a good guess as to whether *f*kc is tending towards FINITE or INFINITE after fewer iterations (lower I parameter value) than are required for making a good guess about an *f*c map.

In the software's default cubic Mandelbrot mode, the user has control of a select cursor in the k-plane, meaning that each cursor position selects a two-dimensional value of $k = a + bi$. Whenever you right-click to select a k, you see Mk, the k-section of the cubic connectedness locus. By slightly varying a and/or b, you can look at k-sections near each other, and try to visualize stacking them one atop the other.

If $k = 0 + 0i$, you get a degenerate Mk with four-fold symmetry; this is the default cubic Mandelbrot set. If $|k|$ is large (about 1 or 2), you get an empty Mk. All the Mk have symmetry around the origin, meaning that if $c = u + iv$ is in Mk, $-c = -u - iv$ will be in Mk, too.

The appearance of the Mk is quite interesting. You often see small replicas of the quadratic Mandelbrot set, though sometimes with wedges cut out of it. (For an example of this, see the preset image labeled "Laugh.") Looking at successive sections, you often get the impression of the Mk being like a cross-section of a three-dimensional Mandelbrot shape, with buds all over it. But of course the full CCM is four-dimensional, and this shows up in the fact that many of the bud cross-sections have pieces missing from them. Presumably these missing pieces have inappropriate four-dimensional coordinates for intersecting Mk.

As an aid to mathematical visualization, think of it this way. The CCM is like a writhing three-dimensional solid, observed over a period of time. If a section of a bud seems to have the right half missing, you might think of the left half of the bud as being in Monday and the right half of the bud as being in Tuesday, with your cross-section being computed at the Monday time coordinate. Time is not used in a physical sense here, but simply for the vividness of the image.

Now for an explanation of how to compute Mk. The computations of M, the Jc, and the Jck are quite similar. The computation of an Mk is done by fixing $k = a + ib$ and then varying $c = u + iv$ and checking whether $k \xrightarrow{fkc} \text{FINITE}$ and $-k \xrightarrow{fkc} \text{FINITE}$. On the basis of what we learn while doing this checking, we assign an Mk color(c) to the pixel representing c. Before starting, we select a BOUND and a value of MAXITERATION, and say that $k \xrightarrow{fkc} \text{FINITE}$ if for all n less than MAXITERATION, the nth image of k is has norm less than BOUND. We always take BOUND to be the square root of 7, and MAXITERATION can be set by the user as the parameter I.

Assuming $k = a + bi$ has been set, the computation of the Mk color(c) for a fixed $c = u + iv$ takes place as follows (this pseudocode description is not fully optimized):

```
initialize:
    iteration = 0;
    tx = a;
    ty = b;
```

```
        a3 = –3*( a*a – b*b );
        b3 = –3*( 2*a*b );
    and then iterate:
        x2 = tx*tx;
        y2 = ty*ty;
        xa = x2 – y2 + a3;
        yb = 2*tx*ty + b3;
        temp = xa*tx – yb*ty + u;
        ty = xa*ty + yb*tx + v;
        tx = temp;
        iteration = iteration + 1;
    until x2 + y2 > BOUND or iteration = MAXITERATION, saving
        iteration as iteration1;
    and then re-initialize:
        iteration = 0;
        tx = –a;
        ty = –b;
        a3 = –3*( a*a – b*b );
        b3 = –3*( 2*a*b );
    and iterate the same computation,
    until x2 + y2 > BOUND or iteration = MAXITERATION, saving
        iteration as iteration2;
    If iteration1 and iteration2 both equal MAXITERATION, then
        Mk color(c) = 0.
     Otherwise,
        Mk color(c) = TABLE[smaller of iteration1 and iteration2].
```

What is TABLE? The problem is that there are only sixteen colors available on the computer screen, and the iteration counts can range over hundreds of possible values, depending on how large MAXITERATION is. The simplest kind of TABLE operation would be to simply let TABLE[n] = n MODULO 16, the remainder obtained from dividing by 16. In practice, however, it is better to use a TABLE that does things like banding the colors across the iteration counts with, say, counts 100 through 110 all getting color 1, counts 111 through 120 getting color 2, and so on. The user can alter the structure of TABLE with the BANDWIDTH variables that are set using the W and Shift+W keys or the Options menu.

Recall that the Mk sets are cross-sections of the four-dimensional CCM which is, once again, the set of all quadruples (a,b,u,v) such that if k = a+ b*i* and c = u + v*i*, then the cubic Julia set Jkc is connected. The Mk are obtained by fixing the values of a and b and looking at the cross-sectional pattern in the uv-plane. You can imagine looking at cross-sections taken in different planes; there are in fact six different ways of choosing two out of four axes. If the slice type is set to the ab-plane instead of the uv-plane, then the cubic Mandelbrots being shown can be thought of as Mc sets, where c = u + v*i* is fixed and you look at the range of k for which Jkc is connected. In other words,

Mc = {k : Jkc is connected}.

By using the L and Shift + L keys or the Options menu, you can look at sections of the CCM taken in each of six different planes. It is an interesting (though perhaps impossible) challenge to flip through the different views and try to imagine how they fit together into a four-dimensional whole.

The last fractal type that the program shows is what we call the Rudy set R. Rudy Rucker invented it as an analogy to the Mandelbrot set M. Since M can also be defined as { c : c is in Jc }, he defined R in a similar "self-referential" or "diagonal" fashion as

R = {k : k is in Mk}
  = {k —*f*kk→FINITE AND -k—*f*kk→FINITE}.

If you think in terms of the Mc sets in the ab-plane you get a variant of the Rudy set, which can be viewed by pressing the L key when the Rudy set is displayed:

R' = {c : c is in Mc}.

Now we can explain the Bullseye and Feathers fill methods for coloring the interiors of the Mandelbrot, cubic Mandelbrots, and the Rudy set.

In the case of the Mandelbrot set, a point c is in the interior of the set provided that |zn| is less than BOUND for the first MAXITERATION members of the sequence obtained by taking $z_0 = 0+0i$ and $z_{n+1} = (z_n)^2 + c$. To do a Bullseye fill, we keep track of the numerical values of the $|z_n|$ and set MINSIZEc equal to the smallest of these values. The color for the pixel corresponding to c is then set equal to TABLE[16000*MINSIZEc]. We multiply MINSIZEc by such a large number so as to amplify the small differences in MINSIZEc between nearby c. To do a Feathers fill, we make a note of the value of n at which |zn| takes on the small MINSIZEc value, set MINSTEPc equal to this n, and give the pixel the color TABLE[MINSTEPc].

In the case of the cubic Mandelbrots and the Rudy sets, we do more or less the same thing, except we look at twice as many complex numbers' distance from the origin to find the relevant MINSIZE.

Chapter 3
Magnets and Pendulum

> *The laboratory mouse of the new science was the pendulum: emblem of classical mechanics, exemplar of constrained action, epitome of clockwork regularity. A bob swings free at the end of a rod. What could be further removed from the wildness of turbulence? . . . Yet the pendulum still had surprises in store.* (Chaos: Making a New Science, *page 39.)*

One of the stranger by-products of the new science of chaos is this: it has become respectable again for a serious theoretical physicist to spend time studying the dynamics of pendulums. Even in this purest of mechanical systems, chaotic behavior can occur.

One simple example is the case of a pendulum swinging above some magnets, all capable of attracting and grabbing onto the pendulum's bob. The bob swings back and forth, around and about, first looping in the ambit of one or two of the magnets, then suddenly jumping to another region. It becomes almost impossible to predict where it will finally settle, until the very last minute. As in so many chaotic systems, the slightest change in the starting point or in the physical setup—the positions and strengths of the magnets, the force and friction on the bob—can entirely change the outcome.

One of the central concepts developed by chaos specialists to understand such systems is the idea of basins of attraction. If a bob is let go from a certain point, and eventually comes to a halt at a certain magnet, then the starting point is in that magnet's basin of attraction. The basin is like the area surrounding a bathtub drain; it is the collection of points drawn toward the drain. Many real systems seem to have basins of attraction, from the weather to the human brain. The astonishing feature of such basins is that they need not form solid regions. If the basins of attraction for a given layout are colored according to which magnet finally captures the bob—red, blue, and green, say—the colors may end up woven together in regions of infinite, fractal complexity.

Things To Try

Begin by trying out some of the prestored combinations of magnets. Press F3 to load each preset layout. Left-click or press ↵ to give the bob a push—the closer the cursor is to the bob, the harder the push. Press H for a new arrangement of magnets, or capture a magnet by moving the cursor over it, and then move the magnet to a new spot. You can change the trail left by the bob by pressing T . You can also try magnets that repel (shown red instead of blue) by pressing − on the numeric keypad. To stop a simulation, clear the trails, and reset the bob to its default location, press R.

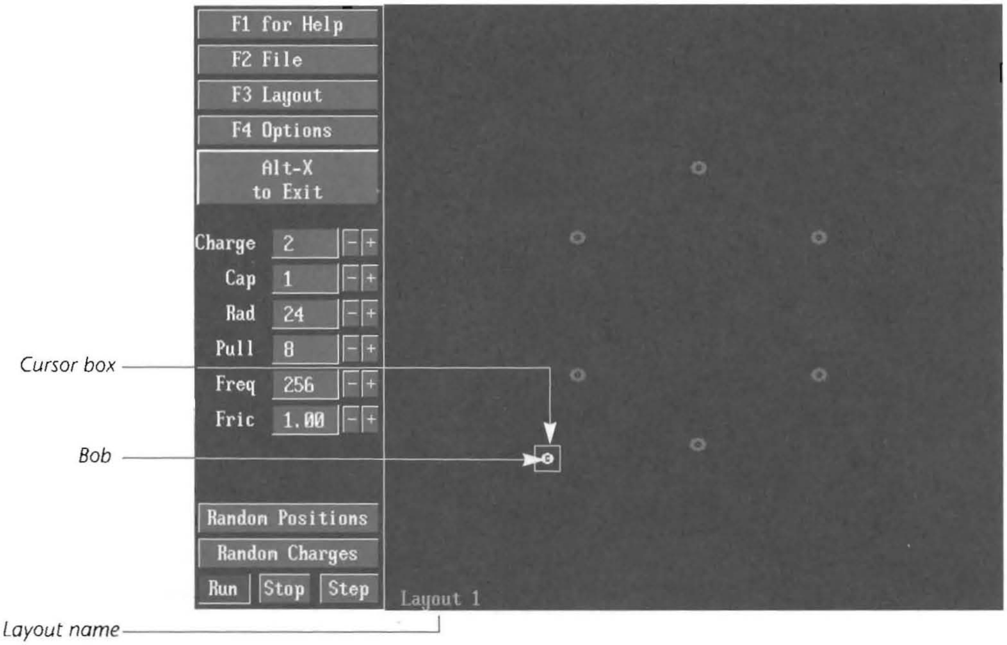
To see an example of basins of attraction, press F3 repeatedly until Layout 7 appears. Press ↵ to start the basins of attraction simulation. The bob moves around the magnets and colored dots appear toward the top of the screen. Gradually, the bob creates regions with mixed colors and jagged edges. These fractal regions demonstrate that you cannot predict where the bob will land in this simulation.

The program must make four passes to fully color in the basins of attraction. Completing the simulation takes several hours. You can observe just the first two passes to get a rough idea of the basins of attraction or you can let the simulation run while you do something else.

To stop the basins of attraction simulation, press J to turn off basins mode, or press F3 to load another preset layout.

Starting the Program

When you start the program, the following screen appears:



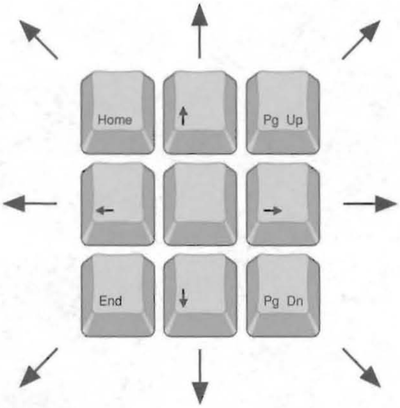
Charge 2
Cap 1
Rad 24
Pull 8
Freq 256
Fric 1.00

The image area of the screen shows a preset layout of six blue magnets and a white pendulum bob inside a box called the *cursor box.* You view a Magnets and Pendulum layout as though you were looking down on it.

The name of the preset—Layout 1—appears in the lower-left corner of the image area.

Moving Around in the Image Area

You can use the mouse or the keyboard to move the cursor around in the image area. To move around using the keyboard, make sure Num Lock is turned off and refer to the following illustration:



Numeric keypad keys and direction of cursor movement

Note: You can press o (the letter "o" key) to toggle between smaller and larger steps for cursor movement.

Experimenting with Preset Layouts

The Magnets and Pendulum program has seven preset layouts. You can display each preset layout and run a simulation that shows the pendulum bob moving around as it is attracted and repelled by positive and negative magnet charges.

As you experiment with the preset layouts and run simulations, you can observe three key aspects of chaotic systems.

- A chaotic system generates complex, highly organized patterns. For example, the pendulum bob might continuously circle around a configuration of magnets, almost repeating its trails, but not quite.
- Though seemingly predictable, these patterns are not predictable at all. For example, you cannot guess with 100 percent accuracy which magnet the bob will stick to, no matter how simple the layout or "normal" the parameter settings.

- Small changes you make cascade upward through the system, producing unpredictable results. This phenomenon is known as *sensitive dependence on initial conditions*. It means that when you make changes to parameter settings (even though you know generally how the changes should affect the simulation or specifically how the equations governing the program's calculations should affect the results), you still will not be able to predict the behavior of the bob.

Note: You can read more about pendulum systems in James Gleick's *Chaos: Making a New Science,* pages 40–44. For more information on sensitive dependence on initial conditions, see *Chaos: Making a New Science,* pages 20–23.

Layout 1

Six positive (blue) magnets form a large circle. After you start the simulation, the bob eventually sticks to one of the magnets.

1. If Layout 1 is not currently displayed, press F3 until Layout 1 appears.
2. Press ↵ to start the simulation running. The bob moves around the screen, attracted by the positive magnets.
3. Press a number key to change the number of magnets in the layout.
4. Press ↵ to start the simulation running again.
5. Press Shift + D to increase the radius of the circle of magnets.
6. Press ↵ to start the simulation running again.
7. Notice that the cursor turns into a yellow diagonal line called the *club cursor.*
8. Move the yellow club cursor to the bob to capture it. The captured bob is enclosed in a small box. (If you capture a magnet by mistake, left-click or press ↵ to release the magnet.)
9. Move the captured bob to a new location and left-click or press ↵ to release the bob. Notice that even a small change in the starting position greatly changes the bob's trajectory.

Note: For more information about running a simulation, see "Running a Simulation" on page 58.

Layout 2

Five negative (red) magnets form a medium-sized circle. Once you start the simulation, the bob can move around the magnets virtually forever. If you let this simulation run long enough, the bob's motion will become quite violent.

1. Press F3 until Layout 2 appears on screen.
2. Press ↵ to start the simulation running.
3. Press V to reverse time direction for the simulation. When you do this, the bob runs backward, erasing all the trails it has left so far.

Reversing the time direction demonstrates that the path of the bob is completely deterministic, although it might look random. The calculations that move the bob to the point where you press V to start reverse can be performed exactly the same in reverse so that the bob erases every trail it makes.

Note: For more information about reversing the time direction, see "Reversing Time Direction" on page 70.

Layout 3

Twenty-four positive (blue) magnets form four rows of six magnets each. In this simulation, the bob cannot stick to the magnets.

1. Press F3 until Layout 3 appears on screen.
2. Press ↵ to start the simulation running.
3. Press T to change the trace type.
4. Press T again to see the next available trace type.
5. Press Alt+R to erase old trails to see only the new trace type.

Note: For more information about trace type, see "Trace" on page 73 and "Trace Erase" on page 73.

You might enjoy a "hunt the butterfly" game with Layout 3. The object of "hunt the butterfly" is to capture the bob without capturing a magnet.

To capture the bob, follow these steps:

1. Press ↵ to start the simulation running. The cursor turns into a yellow diagonal line called the *club cursor.*

2. Try to place the club cursor over the bob. If you successfully capture the bob, you enclose it with the cursor box. If you capture a magnet by mistake, the computer beeps to alert you to the capture.

   You can right-click to release the magnet and send it back to its original location, or you can left-click or press ↵ to release the magnet at its current location.

Layout 4

Sixteen negative (red) magnets form a random, asymmetrical layout. The trail of the bob is set to a fixed length and moves snakelike behind the bob. The parameter settings in this layout make the bob settle into "pockets" or areas on the periphery, rather than sticking to magnets.

1. Press F3 until Layout 4 appears on screen.
2. Press ↵ to start the simulation running.
3. Press H to create a random layout of the magnets.
4. Press ↵ to start the simulation running for the new layout.
5. Press ↵ again to give the bob a push. Watch how the bob's motion partitions the magnets.
6. Press I to assign random charges to the magnets.
7. Press ↵ to start the simulation running.

Note: For more information about random layouts, see "Random Positions" on page 66. For more information about random charges, see "Setting Random Charges" on page 63.

Layout 5

Thirteen negative (red) magnets form a small circle. The parameter settings allow the bob to move through the circle of magnets even though they repel it.

1. Press F3 until Layout 5 appears on screen.
2. Press ↵ to start the simulation running.
3. Press E to change from lively mode to physical mode, and then observe the results when you start the simulation.

4. Press ↵ again to push the bob.
5. Press + to make all magnets positive.
6. Press E to change back to lively mode.

Note: For a description of lively and physical mode, see "Computation" on page 72.

Layout 6

Seven dipole (positive and negative) magnet pairs form a random, asymmetrical layout, and the trail of the bob is set to a fixed length. When you run the simulation, the bob moves from positive pole to positive pole of the dipole magnets. The preset's default parameter settings make it impossible for the bob to stick to any magnet and allow the bob to move around the layout freely without settling in one area.

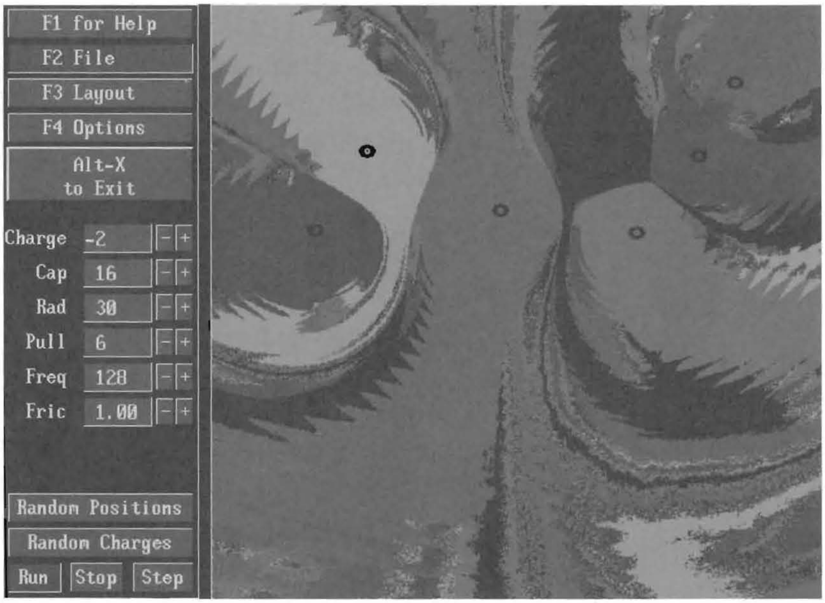
1. Press F3 until Layout 6 appears on screen.
2. Press ↵ to start the simulation running.
3. Move the yellow club cursor over a blue magnet to capture it. The computer beeps to confirm the capture.
4. Press − to change the charge of the magnet you've captured so there is now a pair of negative magnets. Left-click or press ↵ to release the captured magnet.
5. Notice how the bob avoids the pair of negative magnets.
6. Move the yellow club cursor over a red magnet in a dipole pair to capture it. The computer beeps to confirm the capture.
7. Press + to change the charge of the magnet you've captured so that now you have a pair of positive magnets. Left-click or press ↵ to release the captured magnet.
8. Notice what happens to the bob when you run the simulation with two positive magnets next to each other.

Layout 7

Layout 7 demonstrates basins of attraction. Seven different colored, attracting magnets form a random, asymmetrical layout. When you run the simulation, the program creates basins of attraction by repeatedly releasing the bob from a unique starting point and coloring that point the same color as the magnet that the bob lands on.

Although the simulation is set to run as fast as possible, it takes a number of hours to fully color in all the basins of attraction.

1. Press F3 repeatedly until Layout 7 appears on screen.
2. Press ↵ to start the simulation running. The simulation must run for some time before you see the basins of attraction. The completed basins of attraction screen for Layout 7 resembles this:



Charge -2
Cap 16
Rad 30
Pull 6
Freq 128
Fric 1.00

3. To stop the basins of attraction simulation, press J to turn off basins mode, or press F3 to load another preset layout.

When the basins of attraction simulation is complete, the program automatically saves the completed image as a *.gif* file.

Once you've run the simulation for Layout 7, you can change the layout of magnets and run a new simulation. For example, press H to randomize the layout or press D or Shift + D to arrange the magnets in a small or large circle. The new layout produces a completely different set of basins.

Note: For more information on basins of attraction, see "Basins of Attraction" on page 69.

Running a Simulation

When you run a Magnets and Pendulum simulation, you see a system resulting from the unique combination of magnet and bob parameter settings, the layout of the options, and other options such as sound or colors.

Starting the Bob

After you set up a simulation, you run it by starting the bob.

- With the cursor anywhere in the image area, left-click or press ↵.

Once a simulation starts running, the cursor turns into a yellow line called the *club*.

Giving the Bob a Push

To speed up the bob's action temporarily or move the bob to a new area of the simulation, you can give the bob a push with the club. The closer the club is to the bob, the stronger the push will be.

- With the club cursor near the bob, left-click or press ↵.

Capturing the Bob or a Magnet

You can *capture* either the bob or a magnet and move it to a new location. When you capture something, you enclose it with the cursor box.

1. Move the club cursor over the object. Your computer beeps to alert you to the capture of a magnet.
2. Press ↵ to release the captured bob or magnet.

You can also use the mouse to release a captured magnet:

- Left-click to release the magnet where it is.

  or

- Right-click to send the magnet back to its original location.

Slowing Down and Speeding Up a Simulation

Slowing down the speed of a simulation makes it easier to see the movement of the bob. You can slow down or speed up a simulation within a range of speeds. The default is set to maximum speed.

- Press (,) (comma key) to slow down a simulation by 0.1 second per simulation step.
- Press (.) (period key) to speed up a simulation by 0.1 second per simulation step.

Stopping a Simulation

You can stop the bob's movement without making any other change.

1. Press the spacebar or select the Stop button to stop the bob without erasing trails.
2. Press (Shift) and the spacebar or select the Run button to start the bob moving again.

Running a Simulation in Steps

You can run a simulation one step at a time.

1. Select the Step button or press the spacebar to stop a simulation.
2. Select the Step button or press the spacebar repeatedly to run a simulation one step at a time.
3. Select the Run button or press (Shift) and the spacebar to start running the simulation again.

Erasing Trails from a Simulation

Keyboard shortcut: (Alt)+(R)

You might find you want to keep the bob moving but erase the trails once from a simulation. Erasing the bob's trails from a simulation lets you keep closer track of the bob's action around the screen.

- Press (Alt)+(R) to erase the bob's trails from the screen without interrupting a simulation.

Erasing Trails and Stopping the Bob

Keyboard shortcut: S

You can also erase trails and stop the bob at its current location. Then you can start a simulation again from the point where the bob stops.

1. Press S to erase the bob's trails from the screen and stop the bob at its current location.
2. Left-click or press ↵ to continue the simulation from the current location of the bob.

Reset

Keyboard shortcut: R

You can stop a simulation, erase all trails, and return the bob and cursor box to the lower-left corner of the image area (the starting default location).

- Press R to reset a simulation.

Changing Parameters

Changing the parameter settings for a simulation lets you observe how even small changes in initial conditions can dramatically affect the outcome of an experiment with Magnets and Pendulum. You can set new parameters before running a simulation, or you can change parameters while the simulation runs.

Many of the most commonly used parameters appear on the main screen of Magnets and Pendulum.

Note: When you find an interesting set of parameters and magnet layout, you can save the layout in memory or in a parameter file. For more information, see "Saving and Loading Files" on page 74.

Changing the Charge of Magnets

Keyboard shortcut: Shift + C to increase; C to decrease

Each magnet has a numerical positive or negative *charge.* Charge sets the strength of the attracting or repelling force of the magnets.

The Magnets and Pendulum program gives you several ways of changing the charge of the magnets in a layout.

If All Magnets Have the Same Charge (Charge Button)

If all magnets in the current layout have the same charge, that value appears in the Charge field on the main screen.

- Enter a value between **–500** and **500** in the Charge field to change the charge of all magnets in the layout.

You can also use a keyboard shortcut to adjust the charge. Adjust the charge for all magnets like this:

1. Make sure the cursor box is not enclosing a magnet.
2. Press (Shift)+(C) to increase the charge of all magnets or press (C) to decrease the charge.

Adjust the charge for one magnet like this:

1. While running a simulation, move the club cursor to a magnet to capture it.
2. Press (Shift)+(C) to increase the charge of the captured magnet or press (C) to decrease the charge.

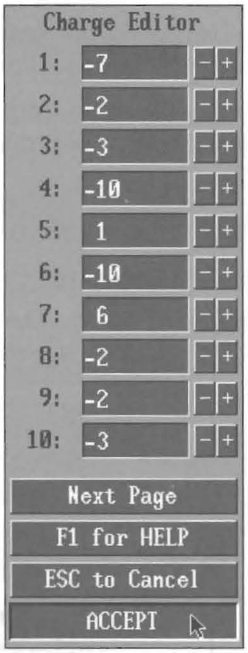
Note: You can also change the charge of one magnet by pressing (F5) to display the Charge Editor. The Charge Editor is described in the following section.

If Magnets Have Different Charges (F5 Charges Button)

If the magnets in a layout have different charges, the F5 Charges button appears in place of the Charge field. The F5 Charges button lets you display the Charge Editor, which lists the individual charge for each magnet in a layout.

To use the Charge Editor, follow these steps:

1. Select the F5 Charges button or press F5. The Charge Editor appears.



Charge Editor

1: -7
2: -2
3: -3
4: -10
5: 1
6: -10
7: 6
8: -2
9: -2
10: -3

2. Move the highlight down the list of charges and watch the white circle move from magnet to magnet in the layout. The white circle lets you match each magnet with its entry field in the Charge Editor.
3. If the current layout has more than 10 magnets, press the Next Page button to see additional magnet charges.
4. When the white circle is on the magnet whose charge you want to adjust, enter a value between **–500** and **500** in the entry field and press ↵ .
5. Select the ACCEPT button to accept the changes and exit the Charge Editor.

If you set large negative or positive charges, you need to increase the center pull force to keep the bob on screen. See "Center Pull Force (Pull)" on page 65 for details. If you lose sight of the bob on the screen, press R to reset the bob's starting position.

Suggestion: To increase your chances of predicting the bob's movement, try an experiment.

1. Select the F5 Charges button to display the Charge Editor.
2. Move the highlight to two or three of the strongest positive charges shown in the Charge Editor dialogue box. As you move the highlight, observe the image area to see which positive magnets the white disk lands on.
3. Press (Esc) to exit the Charge Editor.
4. As the simulation continues to run, observe that, although you can now see that the bob "prefers" the two or three magnets with the strongest positive charge, you still cannot predict precisely where the bob will move next.

Switching Positive and Negative Charge for Magnets

To switch the charge for an individual magnet, follow these steps:

1. Capture the magnet with the cursor box.
2. Press (+) to switch the magnet's charge to positive. Press (−) to switch the charge to negative.

To switch the charge for all magnets, follow these steps:

1. Make sure the cursor box is not enclosing any magnet.
2. Press (+) to switch all magnets to positive charge. Press (−) to switch to negative charge.

Setting Random Charges

Keyboard shortcut: (I)

You can set random charges for all magnets.

- Select the Random Charges button at the lower-left corner of the screen to assign random positive and negative charges. If all magnets had the same charge before you randomized the charge, the Charge field changes to the F5 Charges button.

Suggestion: After setting random charges, you can use the Charge Editor to view or change each magnet's charge. To use the Charge Editor, see "If Magnets Have Different Charges (F5 Charges Button)" on page 61.

Capture Radius (Cap)

Keyboard shortcut: Shift + X to increase; X to decrease

All magnets have the same *capture radius*, which measures in pixels how close the bob can come to a magnet before it sticks. The current capture radius appears in the Cap field on the main Magnets and Pendulum screen.

- Enter a value between **0** and **20** in the Cap field to change the Capture Radius.

Suggestion: To start a simulation that runs endlessly, set the capture radius at 0.

Magnet Radius (Rad)

Keyboard shortcut: Shift + K to increase; K to decrease

Magnet radius shows in pixels the density of all magnets' charges. Setting a small magnet radius means that the magnets behave like very intense, point-like charges. Setting a large magnet radius means that the magnets act like gentle, diffuse charges. CHAOS displays the current magnet radius in the Rad field on the main screen.

- Enter a value between **1** and **60** in the Rad field to change the Magnet Radius.

To set the force at the magnets' centers to zero, CHAOS requires that when bob-to-radius distance is less than magnet radius, the force must be proportional to charge multiplied by distance. You can find complete information on calculations for magnet radius in "The Mathematics of the Program" on page 74.

Suggestion: If you set a small magnet radius, you will probably want to set sampling frequency fairly high to keep the bob from moving too rapidly and erratically around the screen. See "Sampling Frequency (Freq)" on page 65 for details.

Center Pull Force (Pull)

Keyboard shortcut: (Shift)+(B) to increase; (B) to decrease

Center pull force shows the strength of the force pulling the pendulum bob toward the center. When you change this parameter, it's as though you're controlling the size of the restoring force that drives the pendulum. The current center pull force appears in the Pull field on the main screen.

- Enter a value between **–500** and **500** in the Pull field to change the Center Pull Force.

Sampling Frequency (Freq)

Keyboard shortcut: (Shift)+(G) to increase; (G) to decrease

CHAOS updates the bob's position step-by-step in the program's own simulation time. Each step takes the same amount of real time to calculate. *Sampling frequency* measures the number of real-time steps the program performs per second of simulation time. When you change sampling frequency, you affect how slowly or rapidly the bob moves around, and how smooth or jagged the bob's trail looks. The current sampling frequency appears in the Freq field on the main screen.

- Enter a value between **2** and **10000** in the Freq field to change the Sampling Frequency.

Suggestion: If you want a smoother trail, set a high sampling frequency. When you do, CHAOS divides each second of simulation time into many calculation steps and makes the bob move around the screen slowly with a smooth trail. Conversely, if you want a more jagged trail, set a low sampling frequency. When you do, CHAOS skips over a lot of simulation time for each real-time step and makes the bob move around the screen rapidly with a jagged trail.

Friction (Fric)

Keyboard shortcut: (Shift)+(F) to increase; (F) to decrease

The bob has a numerical *friction*, which measures the force slowing down the bob as it moves. The current friction appears in the Fric field on the main screen.

- Enter a value between **0** and **500** in the Fric field to change the friction of the bob.

Note: CHAOS follows conventions of aerodynamics and calculates friction as proportional to the square of the bob's speed. You can find complete information on these calculations in "The Mathematics of the Program" on page 74.

Changing the Layout of Magnets

Another way of changing a simulation is to change the layout of the magnets. You can arrange magnets in a symmetrical layout such as a circle, you can move individual magnets to any location on screen, or you can use random layouts. You can also change the number of magnets in a layout or add or delete individual magnets.

Note: If you find a layout you like, you can save it temporarily in memory by pressing (N). To load it again, press (F3) repeatedly until that layout appears on screen. For more information, see "Saving and Loading Files" on page 74.

Random Positions

Keyboard shortcut: (H)

To position magnets randomly

- Select the Random Positions button at the lower-left corner of the screen.

Displaying Symmetrically Arranged Magnets

You can use the number keys to display nine different symmetrically arranged magnet layouts. For example, if you press (4), CHAOS arranges four magnets in a square. If you press (7), CHAOS arranges seven magnets in a circle. If you press (1), a single magnet is centered on screen.

To display a symmetrical layout

- Press a number key from (1) through (9) to display a symmetrical arrangement of the corresponding number of magnets.

Suggestion: If you press (0) to remove all magnets from the screen, you can run a simulation and watch the bob's center-pulling pendulum force at work without interference from magnetic attraction or repulsion.

Moving Magnets

You can change the location of one or more magnets in any layout:

1. With a simulation running, move the club cursor to a magnet to capture it. The computer beeps and the cursor box encloses the magnet.
2. Move the magnet to the new location and left-click or press (↵) to release it.
3. Repeat steps 1 and 2 for as many magnets as you want to move.
4. Press (R) to stop the simulation and reset the bob to its starting position.

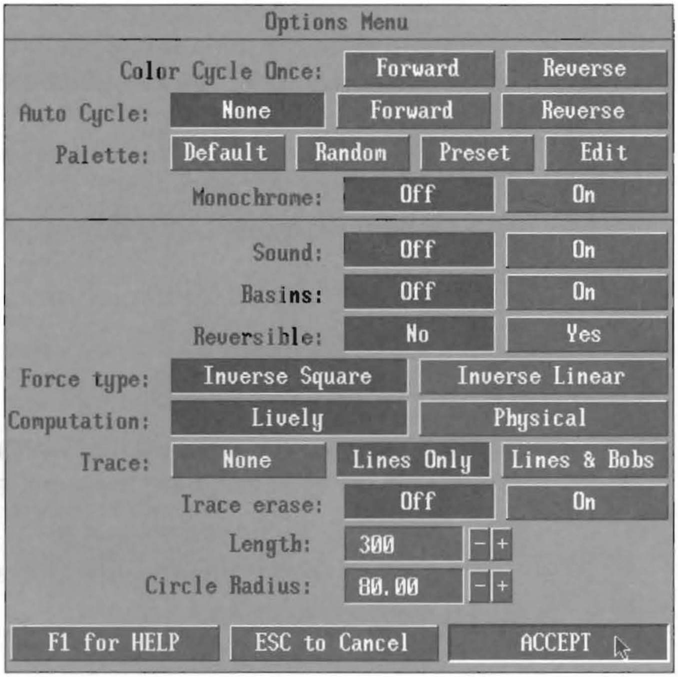
Inserting and Deleting Magnets

You can insert and delete magnets at the cursor location. You can display as many as 31 magnets on screen, or as few as 0.

- Press (Ins) to insert a magnet at the club cursor location.
- Capture the magnet with the cursor box and press (Del) to delete a magnet.

Using the Options Menu

You can use the Options menu to change colors, turn on sound, or start a simulation showing the basins of attraction for a configuration of magnets. To display the Options menu, select the F4 Options button:


Options Menu
Color Cycle Once: Forward Reverse
Auto Cycle: None Forward Reverse
Palette: Default Random Preset Edit
Monochrome: Off On
Sound: Off On
Basins: Off On
Reversible: No Yes
Force type: Inverse Square Inverse Linear
Computation: Lively Physical
Trace: None Lines Only Lines & Bobs
Trace erase: Off On
Length: 300 - +
Circle Radius: 80.00 - +
F1 for HELP
ESC to Cancel
ACCEPT

Color Options

The following color options are common to five of the six CHAOS programs and are described in "Using Color Options" on page 214.

- Color Cycle Once
- Auto Cycle
- Palette (Default, Random, Preset, Edit)
- Monochrome

Sound

Keyboard shortcut: Alt+V

You can turn on sound in Magnets and Pendulum to hear how chaotically the bob is behaving in the current magnet configuration. As the movement of the bob becomes more erratic, the sound also becomes more erratic. And the sound's pitch goes up or down as the bob moves faster or slower.

- Select the On or Off button next to Sound on the Options menu to turn sound on or off.

Note: CHAOS temporarily turns sound off whenever you move the mouse pointer into the menu area.

Basins of Attraction

Keyboard shortcut: J

To illustrate the unpredictability of the bob's attractions to magnets, you can turn on a feature called *basins of attraction*. Basins of attraction show the patterns of colors the program makes when the bob is released from one screen position to the next, with each starting position colored like the magnet the bob lands on.

CHAOS fully colors in the simulation in four passes. First, the program tests every eighth pixel, then every fourth, then every second, and finally every pixel. The fully colored-in simulation has regions of solid colors, but it also has regions of intermixed colors, as well as regions where colors have jagged, or zipper-like, boundaries. The regions with mixed colors and jagged boundaries tend to be fractals and demonstrate that you cannot completely predict where the bob will land. To find out the details of the calculations in basins of attraction, see "The Mathematics of the Program" on page 74.

When the basins of attraction simulation is complete, the program automatically saves the completed image as a *.gif* file. The program names these *.gif* files MAGNETnn.GIF with "nn" representing a two digit number.

Although CHAOS sets the basins of attraction simulation to run quickly, running the simulation might take a number of hours. You can let the simulation run overnight, or you can observe just the first two passes that CHAOS calculates. These two passes give you a rough idea of the basins of attraction for a given configuration of magnets.

To increase the speed of the basins of attraction simulation, you can make the following changes:

- Set the capture radius above 10. Use the Cap field on the main screen to make this change.
- Set the sampling frequency to about 100. Use the Freq field on the main screen to make this change.
- Make sure you're running the simulation at maximum speed. Press . (period key) several times to increase the simulation speed to its maximum.

You can set up a new configuration of magnets for the basins of attraction simulation, or you can use Layout 7. See "Layout 7" on page 56 for details on this preset layout showing basins of attraction.

To set up, turn on, and start a new basins of attraction simulation, follow these steps:

1. Set up a configuration of any number of positive magnets.
2. Select the On button next to Basins on the Options menu or press J and then press R to turn on basins and display each magnet with a unique color.
3. Press ↵ to start the simulation.

Suggestion: To focus on where the bob is landing in the basins mode, slow down the simulation by pressing , (comma key) several times. Watch the bob a while, paying special attention when it colors in areas with mixed colors and jagged edges. Speed up the simulation again by pressing . (period key) several times.

Reversing Time Direction

Keyboard shortcut: U reversible toggle; V reverse

When you reverse time direction for a simulation, you see the bob's trail running backward. This gives you an "instant replay"—in reverse—for the simulation you just saw. Reversing time direction works correctly only if reversible mode is on and friction is set to 0. The default is reversible mode off.

Reversing time direction involves several steps: first, you turn friction off, next, you turn reversible mode on, and finally, when you're ready to see the bob's trail running backward, you start reverse. You can turn reversible mode on any time while you are running a simulation. If you like, you can observe reverse time direction in several simulations; whenever you're ready to stop doing that, you can turn reversible mode off again.

Note: For best results with Reversible mode, use the Lively Computation option and a frequency (Freq) no higher than 300. You might find it easier to see the effects of reversing time direction if you press R to erase trails and reset the simulation first.

To reverse time direction, follow these steps:

1. Enter **0** in the Fric field on the main screen to turn friction off.
2. Press U or select the On button next to Reversible on the Options menu to turn on reversible mode.
3. If you don't have a forward-running simulation going, left-click or press ↵ to start a forward-running simulation.
4. When you're ready to reverse time direction, press V to start the bob's trail running backward.

   The bob's trail runs backward to the point where you pressed ↵ to start the simulation. When it reaches that point, it has velocity zero, just as it did when it started, so it repeats the same path. You can reverse the bob again by pressing V.
5. When you're ready to stop observing reverse time direction for the bob's trails, press U to turn off reversible mode.

Force Type

Keyboard shortcut: A

Magnet force is one of two types of simulated physical law that the magnets obey: Inverse Square force or Inverse Linear force. The default force type is Inverse Square.

When you select Inverse Square force, CHAOS sets magnet force proportional to one over the bob-to-magnet distance squared. When you select Inverse Linear force, CHAOS sets magnet force proportional to one over the bob-to-magnet distance. You can read more about these calculations in "The Mathematics of the Program" on page 74.

- Select the Inverse Square button to use inverse square force.
- Select the Inverse Linear button to use inverse linear force.

Computation

Keyboard shortcut: E

CHAOS lets you change the way the program calculates the movement of the bob. The simulation works by repeatedly calculating the net force on the bob, and then moving the bob and calculating the net force at the bob's new location. In one way of calculating the net force—when the program is running in *lively mode*—the calculation looks only at the bob's current location. In physics, this method of calculation is called the Euler method. When you run the program in lively mode, you get a less accurate, faster calculation of the bob's movement, and the bob's movement becomes quick. The default method of calculating how the bob is likely to move is the lively mode.

In the other way of calculating the movement of the bob—when the program is running in *physical mode*—the calculation of the net force examines the force at several points near the bob's location and takes a weighted average. In physics, this method of calculation is called the Runge-Kutte method. When you run the program in physical mode, you get a more accurate, slower calculation of the bob's movement, and the bob's movement becomes smoother.

For complete details on the calculations for lively and physical modes, see "The Mathematics of the Program" on page 74.

- Select the Lively button next to Computation on the Options menu to use lively mode.
- Select the Physical button next to Computation on the Options menu to use physical mode.

Trace

Keyboard shortcut: T to cycle through trace and trace erase types

You can choose from three different *trace types*, or types of trails for the bob:

- Select the None button if you want no trail behind the bob.
- Select the Lines Only button to see a thin line behind the bob.
- Select the Lines & Bobs button to see a trail of lines and bob images.

Trace Erase

Keyboard shortcut: T to cycle through trace and trace erase types

The Trace Erase option lets you change how the trace type is painted on screen. With Trace Erase off, the trace type (Lines or Lines & Bobs) is shown on screen as one continuous line that traces the bob's path from the start to finish of the simulation. If a simulation runs long enough, the screen can get very crowded with lines. With Trace Erase on, the trace type appears as a constantly moving fixed-length line. (The length of the line is determined by the value of the Length option.)

- Select the On or Off button next to the Trace Erase option to turn trace erase on or off.

Length

Keyboard shortcut: Shift + L to increase; L to decrease

You can change the length of the displayed trail. The program measures trace length as a number of successive calculated bob positions, connected by a line. The default length is 300.

- Enter a value between **0** and **512** in the Length field on the Options menu to change the trace length.

Suggestion: Setting a long trace length, such as 200, makes it easier to see where the bob is spending a lot of time. Setting a short trace length, such as 30, makes the trail appear like a ribbon waving behind the bob.

Circle Radius

Keyboard shortcut: Shift + D to increase; D to decrease

The Circle Radius option lets you increase or decrease the radius of a circle of magnets in a layout. The current radius (measured in pixels) appears in the entry field next to Circle Radius on the Options menu.

- Enter a value between **1** and **135** to change the radius for a circle of magnets.

Saving and Loading Files

You can use the options on the File menu to save and reload parameter files and *.gif* files. For more information, see “Saving and Loading Files” on page 209.

In Magnets and Pendulum, you can also save a layout temporarily in memory. When you do this, CHAOS appends the layout to the end of preset layouts. You can save up to 32 layouts in memory. You can reload the temporarily saved layout if you have not exited Magnets and Pendulum to the Main menu or quit CHAOS to DOS.

To save a layout temporarily in memory

- Press N. The computer sounds two beeps to alert you that the layout has been saved in memory.

To reload a layout temporarily saved in memory

- Press F3 repeatedly until you find the layout you want.

The Mathematics of the Program

In Magnets and Pendulum, we track the motions of a moving bob. The simulation works by keeping track of the bob’s position and its velocity, repeatedly updating the velocity on the basis of the forces felt by the bob, and updating the position on the basis of the velocity. To simplify our calculations, the bob has a mass of one, and the distances are measured in pixels.

There are three kinds of forces: “magnetic forces,” a pendulum force, and the force of friction.

The phrase "magnetic forces" could be replaced by "gravitational forces" if we think of the "magnet" disks as "planets." More logically, we should really speak of "electrostatic forces" and regard the "magnets" as "charged particles." This makes sense because we want to have repelling forces as well as attractive ones. If we accept the electrostatic model and think of the bob as bearing a fixed charge of minus one, then it is attracted by the positively charged disks and repelled by the negatively charged ones. For the rest of this section, let's use the phrase "charge forces" to describe the forces that the charged disks exert on the bob.

The size of the charge force that a disk with charge C exerts on the bob depends on the distance *r* between the center of the disk and the center of the bob. The charge force's magnitude F is given as either

$$F = k*C / r^2 \tag{1}$$

or

$$F = l*C / r \tag{2}$$

depending on whether the force type is set to be inverse square or inverse linear. The numbers k and l are built-in proportionality constants; they are empirically tweaked "magic numbers" that make our simulations look good for the ranges of C and *r* encountered. Since distances are measured in pixels, the *r* values in the denominator are usually rather large, so it turns out that large values of the built-in constants k and l are needed in the numerator. As it happens, our k is 50,000,000.

One problem with the formulas (1) and (2) is that they give very large values when *r* gets close to zero. Yet we feel intuitively that if a metal bob gets near the center of an attracting magnet, it will tend to settle down there, rather than being tossed about by infinitely large forces. Or, again, if one were to fall down a tunnel that goes right through the Earth, one would come to rest at the Earth's center, floating there subject to no gravitational force at all.

To clarify the ideas, suppose that we're looking at a planet with radius R and mass C—and say that the planet is gas instead of rock so we can fall "into" it. Now suppose that we have a unit mass bob at a distance r from the planet's center, with r less than R. In 1690, Isaac Newton used his newly invented calculus to establish that in this situation the bob will be pulled towards the planet's center with a force with magnitude F given as

$$F = k*C_r / r^2 \tag{3}$$

where C_r is the mass of that part of the planet which is within r of its center, and k is a proportionality constant as before. If we suppose that the planet is uniformly dense, then mass is proportional to volume. A sphere with half the radius of another sphere has one-eighth the volume, a sphere with one-third the radius has one-twenty-seventh the volume, and so on. The mass C_r forms a sphere with a volume (r/R) as big as the full sphere, so we expect that

$$C_r = C*(r/R)^3 \qquad (4)$$

If we plug this expression for C_r into formula (3) and simplify, we get

$$F = k*C*r / R^3 \qquad (5)$$

This is a nice formula because it makes F go down to zero as r goes to zero, which is what we wanted in the first place.

The quantity R is what the Magnets and Pendulum program calls "magnet radius." Even though our screens show bob and magnets as two-dimensional disks of fixed size, the program actually acts as if the bob were a point, and as if the magnets were small, three-dimensional spheres which the bob is capable of moving through. In the inverse square case, when the bob is further than R from a magnet, the force's magnitude is calculated using formula (1), and if the bob is less than R pixels from a magnet's center, the force's magnitude is obtained by using formula (5). A similar trick is used in the inverse linear case.

Unlike the charge forces, which get weaker with increasing distance, the pendulum force gets stronger as the bob moves away from the center of the screen. Instead of thinking of a pendulum, you can also think of a spring with one end attached to the screen center and the other end attached to the bob. The further you stretch the spring, the harder it pulls back. The intensity of the pendulum force is set by means of a center pull parameter P. You can think of P as being the strength of the gravitational force which drives the pendulum or, in terms of a spring, you can think of P as controlling the stiffness of the spring. If *r* is the distance from the center of the bob to the center of the screen, then the magnitude of the pendulum force is

$$F = m*P*r \qquad (6)$$

where m is a built-in proportionality constant (it's actually 10).

The frictional force depends not on the bob's position, but on the bob's speed—actually on the square of the bob's speed. The faster the bob moves, the greater is the friction which it encounters; the principles of aerodynamics suggest that the friction, or drag, is in fact proportional to the square of the speed. The simulations use an adjustable friction parameter D. If the bob's present speed is *V*, the drag due to friction has magnitude equal to

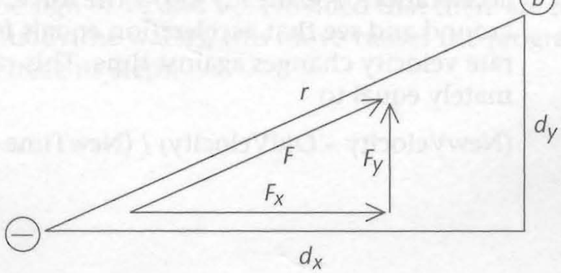
$$n*D*V^2 \tag{7}$$

where n is a built-in proportionality constant (we use 0.01) designed to give good results for the ranges of F and V that we typically use.

As mentioned before, the simulation operates by keeping track of the bob's position and velocity. A complicating factor is introduced by the fact that velocity is a vector: moving left twenty pixels per update is not the same as moving down twenty pixels per update. Just as position is an ordered pair of position coordinates (x,y), velocity is an ordered pair of speeds (v_x,v_y). A velocity of (–10,20) might mean that the bob wants to move ten pixels left and twenty pixels up during the next update. The total speed V can be obtained from the velocity by using the Pythagorean theorem

$$V^2 = v_x{}^2 + v_y{}^2.$$

The charge forces, pendulum force, and friction forces are all vectors as well. That is, each of the forces is really a vector (F_x,F_y). The force formulas determine, F, the magnitude of the whole vector, rather than the actual components F_x and F_y. How do we break F into F_x and F_y? The solution depends on the fact that for each of the three kinds of forces, we know what direction the force vector is supposed to point in. A magnet's charge force points along the line between the bob and the magnet's center. The pendulum force points along the line between the bob and the center of the screen. And the retarding force of friction points in the opposite direction from the bob's present velocity.



Consider, for instance, the case where the bob is being repelled by a negative magnet charge. Imagine drawing a line from the bob to a magnet's center, a line with length r. We can break this line into a horizontal and a vertical component by making it the hypotenuse of a right triangle with a horizontal side and a vertical side. Call the respective lengths of the two sides d_x and d_y, and note that d_x is the difference between the bob's x-coordinate and the magnet's x-coordinate, while d_y is the difference between the bob's y-coordinate and the magnet's y-coordinate. r is actually calculated by using the Pythagorean formula $r^2 = d_x^2 + d_y^2$. Now, the force F which the magnet exerts on the bob will be directed from the bob away from the magnet along the line connecting their centers. And if we break F into horizontal and vertical components of size F_x and F_y, these component lines will be parallel to the lines representing dx and dy. Since all three pairs of sides are parallel, the triangles are similar, so we can know that their sides bear the same ratios to each other. F_x is to F as d_x is to r; and F_y is to F as d_y is to r. If we write the ratios as fractions, set them equal to each other, and multiply through by F, we get the formulae:

$$F_x = F*d_x / r \text{ and } F_y = F*d_y / r \qquad (8)$$

Using formulae like (8) with formulae for F enables us to calculate the forces in the x and y directions. If, for instance, we combine (8) with (1), we get

$$F_x = k*C*d_x / r^3, \text{ and } F_y = k*C*d_y / r^3 \qquad (9)$$

In the simulation, we first sum up the F_x's and the F_y's contributed to the bob by each of the magnet computations. If there are four magnets, for instance, then F_x is computed as $F_{xm1} + F_{xm2} + F_{xm3} + F_{xm4}$. Next the F_{xc} and F_{yc} due to the center pull are computed and added in. Equation (8) works for center pull as well, provided that one views the pendulum center as the "magnet" at the origin. And the friction? We don't put in the friction until the last step of the velocity update.

Isaac Newton's second law of motion says that force is equal to mass times acceleration. Since the bob's mass is one, force equals acceleration. We already know the force, so we turn the equation around and see that acceleration equals force. Acceleration is the rate velocity changes against time. This rate of change is approximately equal to

(NewVelocity – OldVelocity) / (NewTime – OldTime)

The change in time per simulation step is controlled by the Freq parameter; that is, NewTime – OldTime is equal to 1 / Freq. Thus we get this approximating equation for the x component of the velocity (and a similar one for y):

$$V_x = V_x + F_x / \mathrm{Freq} \tag{10}$$

This method of computing the new velocity components is called the "Euler approximation," and it is the method used by our "lively" mode. There is a slower and more accurate algorithm called the "Runge-Kutte method," which is the what we use in our "physical" mode. The Runge-Kutte method works by evaluating F_x and F_y at four carefully selected points near the bob's current position and averaging the values to get a better approximation to the force which the bob will experience along the entire course of its next motion step.

Once New V_x and New V_y have been calculated, friction is applied. Friction is proportional to the square of the speed V, so to get the final value of V_x, we use this formula (and a similar formula for V_y):

$$V_x = V_x - V_x * m * D * V^2 / \mathrm{Freq} \tag{11}$$

Now that the new values of V_x and V_y are calculated, we get the bob's new position from the equations:

$$x = x + V_x / \mathrm{Freq}, \text{ and } y = y + V_y / \mathrm{Freq} \tag{12}$$

For increased speed, most of our program variables are long integers instead of floating-point numbers. Computers without math coprocessors do integer operations substantially faster than floating-point operations. Of course, rounding off V_x, V_y, x, and y introduces inaccuracies. To prevent this, the program usually keeps track of the total quantities rounded off in floating-point "debt" variables, adding or subtracting a unit whenever a debt gets larger than one or smaller than minus one. This debt-minding process is turned off when the program is put into "reversible mode" by pressing Shift + V and setting the friction to zero. When you press V to reverse the bob's motion, the program simply changes the sign of V_x and V_y. Provided that there are no debt variables to muddy the waters, this move causes the program to precisely retrace its steps.

Chapter 4
Strange Attractors

. . . The strange attractor began as a mere possibility, marking a place where many great imaginations in the twentieth century had failed to go. Soon, when scientists saw what computers had to show, it seemed like a face they had been seeing everywhere, in the music of turbulent flows or in clouds scattered like veils across the sky. Nature was constrained. Disorder was channeled, it seemed, into patterns with some common underlying theme. (Chaos: Making a New Science, *page 152.)*

A strange attractor is a diagram that shows how a chaotic system changes with the passage of time. The system may be as simple as a mechanical pendulum or as complex as an epidemic or a financial market. At each instant in time, a new point is plotted, and the position of that point depends on certain variables that sum up the state of the system—perhaps the velocity of the pendulum, or the percent of the population afflicted by the epidemic, or the price of a stock. The behavior of the diagram, often revealing surprising patterns within disorder, gives scientists a way of understanding the behavior of the full dynamical system.

The starting point of the attractor may be any plausible state of the system—the pendulum set swinging with a certain push. (Often you can choose a starting point by left-clicking the mouse at any place on screen.) The trail of points then heads toward the attractor. It is sucked in—attracted—showing how the dynamical system itself falls into its preferred behavior. The long-term path of the points shows whether the system is orderly or chaotic. If a spreading disease, for example, eventually reaches a certain level, then its attractor will simply be a point. If a system finds a regular oscillation, like a pendulum swinging back and forth in a clock, its attractor will be a simple loop. But many systems turn chaotic, never settling down at all, and their attractors are called "strange." A strange attractor is infinitely complicated because the path never reaches the same point twice. If it did, it would begin to repeat itself.

The mathematical physicist David Ruelle, with Floris Takens, coined the name in 1971, before he had ever actually seen a strange attractor. Later he discovered an "esthetic appeal" beyond their mathematical beauty: "These systems of curves, these clouds of points suggest sometimes fireworks or galaxies, sometimes strange and disquieting vegetal proliferations. A realm lies there of forms to explore, and harmonies to discover."

The Lorenz Attractor

Perhaps the first and most famous strange attractor was discovered by Edward Lorenz, a meteorologist at the Massachusetts Institute of Technology, in 1963. Lorenz made a simple model of the weather to help himself understand why this particular dynamical system was so unpredictable. With the help of many hours of computer time and considerable mathematical intuition, the best picture he could produce was a very simple image of a curving line.

He sensed that the shape—a double spiral in three dimensions, bent like the wings of a butterfly—must follow an infinitely intricate path, but this was hard to imagine, in an era when the word *fractal* had not been invented. He apologized. "It is difficult to reconcile the merging of two surfaces, one containing each spiral, with the inability of two trajectories to merge," he wrote. "We see that each surface is really a pair of surfaces, so that, where they appear to merge, there are really four surfaces . . . and we finally conclude that there is an infinite complex of surfaces."

Decades passed before computer visualization finally made it easy to see the full richness of Lorenz's attractor. It exhibits another quality that is characteristic of strange attractors. The tiniest change in the starting point quickly grows to become a great divergence in the outcome. Two paths that start nearby quickly pull apart. In the Earth's weather, that tendency is the surprising phenomenon that Lorenz named the Butterfly Effect: an atmospheric perturbation as tiny as the flapping of a butterfly's wings can cause large transformations in the weather across the globe within about a month. Hence the inevitable unpredictability of the weather over the long term.

Things To Try

The screen shows a "flock" of paths, and you can change the flock to one, or a few, or many (press V). Notice how nearby paths pull apart. You can see the paths with or without little "airplanes" (press Ins for each airplane) that help show the three-dimensionality of the movement. You can use the cursor either to launch new paths or to zoom in (press E to select which of these will be active, and left-click to launch the path or start the zoom).

Yorke's Attractors

The mathematician James Yorke—who gave the young field of chaos its name—has explored an unusual kind of attractor, giving rise to images with an uncanny resemblance to wavy or gnarled forms in nature. They illustrate the power of a phenomenon called "mode locking," or quasiperiodicity, to pull chaos into new patterns. Mode locking is what causes two pendulum clocks that are touching each other to start swinging in synchronization; and causes the moon to lock into an orbit in which the same face always points toward the earth.

Things To Try

Slight adjustments cause remarkable changes in the pattern. First, press F3 to display the Types menu and select Yorke on the menu. Then you can start exploring by pressing N to see some of the rich possibilities.

Hénon's Attractors

Around the time Lorenz was starting to use computers to see chaos in the weather, a French astronomer, Michel Hénon, was doing the same for the stars. Two kinds of attractors bear his name.

The simplest (the Hénon Horseshoe) is a crescent or horseshoe shape that can be imagined as the result of repeatedly folding and stretching, folding and stretching the shape that holds the attractor. As a pastry chef creates a finely layered mille-feuille by folding and stretching the dough, this astronomer created a strange attractor with infinitely many layers. Zooming in on what appears to be two paths shows that the two are really four . . . and then the four prove to be eight, and so on.

Hénon's other kind of attractor (the Hénon attractor) shows how regions of order and chaos can mix in a single space. It grew out of his investigations of the way stars orbit around the center of the galaxy. Imagine that a single star passes from the front of the computer screen into the screen and out the back, leaving a single point where it intersected the screen. Now it orbits around the galactic center, off to the side, and comes back around again. If its orbit is simple, it will strike the same point again. If it is somewhat more complicated, however, its intersections with the screen will start to form patterns—perhaps ovals, or groups of ovals. (Think of the screen as a slice through a doughnut.) If the orbit is completely chaotic, the points of intersection will scatter over a wide region of the screen. By releasing many stars at once, a "flock" of paths, we can see the interweaving of order and chaos, stability and instability. Such patterns also arise in many earthly engineering problems involving regions of different energy levels.

Things To Try

Press F3 to display the Types menu and select Hénon on the menu. Explore the regions of order and disorder in Hénon by starting new paths with the select cursor (left-click or press Ins to start the new path). Each new orbit creates a region of color that may spread across the screen in looping patterns or in scattered chaos.

Next, change to the Hénon Horseshoe attractor. Press F3 to display the Types menu and select Hénon Horseshoe on the menu. Zoom in on Hénon Horseshoe by first pressing E to change to the zoom cursor and then left-clicking or pressing Ins.

The Logistic Attractors

The most fundamental example of chaos may come from an equation that describes a highly idealized wildlife population—for example, fish in a pond.

The idea is that when there are few fish, they are free to multiply rapidly. When the pond starts to fill up, the population growth starts to slow. And when there are too many fish for the available food supply, the population growth changes depending on how big the population already is.

A graph of the equation looks like an arch—the "hump" that appears on screen when you select Logistic Hump on the Types menu. If you imagine that low values of the population are on the left and high values on the right, the steepness of the arch shows what happens to the rate of growth. It is steepest where the population is smallest. When the population is at a middle value, the curve is flat—population growth levels off. When the population is high, the slope is negative, because population growth is negative.

There is just one variable added to the equation, to make things interesting. That is a parameter called *chaoticity,* the parameter you raise or lower by pressing uppercase or lowercase (A), (B), or (C) keys. In the case of a fish population, you might think of this as the general fertility of the fish. You can see the effect of changing this parameter on the shape of the hump. When the chaoticity is low, so is the hump; as it rises, the hump becomes steeper.

The crucial question is what will happen to the population year after year. The answer turns out to depend on the chaoticity parameter.

You might expect that a population will rise each year until it reaches an equilibrium. Perhaps it will overshoot the equilibrium a little, fall back, and eventually zero in on the stable value. Certainly that is what ecologists traditionally assumed. And for low levels of chaoticity, that is exactly what happens.

You can watch the year-to-year change in population by choosing Logistic Pulse on the Types menu. The line traces the rising or falling population like the line of a stock-market chart.

As the chaoticity—the fertility of the fish—increases, something unexpected happens. At a certain point, instead of settling down to an equilibrium, the population goes unstable in a very specific way. It starts to rise and fall in alternate years—oscillating between two different levels.

Continue to increase the chaoticity, and there is another point of crisis—another bifurcation. The two levels divide again, and now there is a cycle of four population values. Then they divide again, and again, faster and faster, through 8, 16, 32 . . . and then, suddenly, a new kind of behavior appears. The population *never* repeats itself. There are no cycles at all—just pure chaos.

This phenomenon, studied using the very simple equation reproduced here, led to an explosion of chaos research. The physicist Mitchell Feigenbaum discovered extraordinary, intricate structure in the pattern of successive bifurcations and developed a theory to explain them. The structure he found has reappeared in many different chaotic systems, from electronic circuits to the human heartbeat.

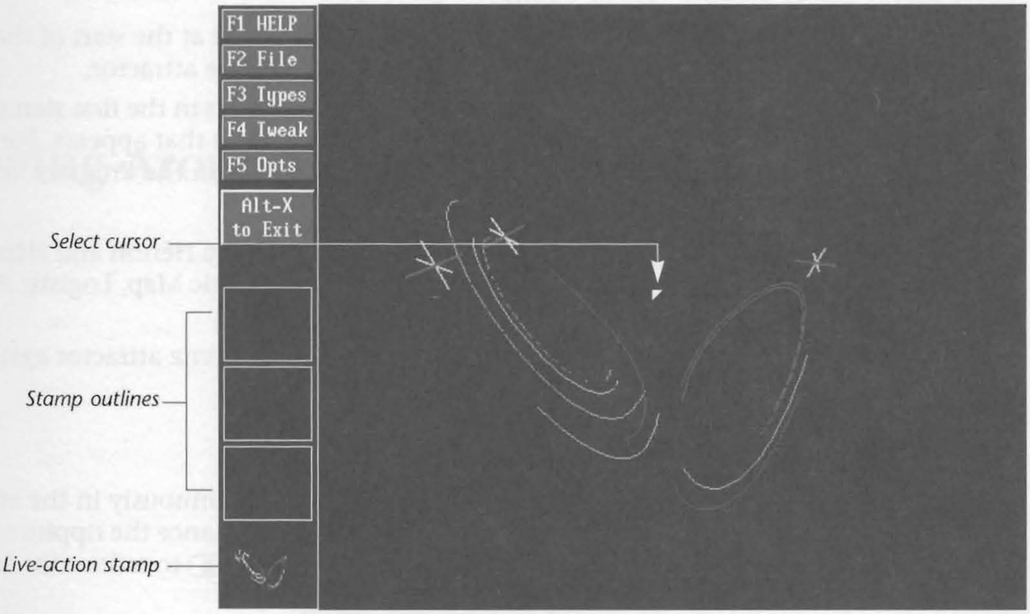
There are three ways of viewing the phenomenon, each related to the others:

- The Logistic Pulse shows the year-to-year change in the most straightforward way.
- The Logistic Hump shows how the curve of the equation works to generate the year-to-year values. Imagine that you start with a population of X and that there's a scale running from left to right across the bottom of the screen. You find X across the scale. Now you draw a line upward until it hits the curve. You determine the new population value from the height at that point. Now the new value is fed into the same equation in a continual feedback loop. This feeding-in of the new value is represented graphically by a sort of short cut. You draw a horizontal line until it hits the diagonal, and then a new vertical line up (or down, as the case may be) to the curve. And so on . . . The trajectory either zeros in on one stable population level, orbits between two or more levels, or runs madly through a whole chaotic sequence of values.
- The Logistic Map is a sort of bird's-eye view of the whole range of chaoticity values, from low (on the left) to high. The level of the population is represented vertically on screen. You can see that as chaoticity increases, the equilibrium level of the population also increases, for a while. But suddenly the equilibrium divides into two levels, one low and one high. Then there are new bifurcations, and then chaos—population values running up and down the screen wildly. Yet even within the zone of chaos, a surprising structure appears. You can zoom in on this zone with the zoom cursor by left-clicking or pressing (Ins). You can

see new cycles—places where there are 3, or 5, or other odd numbers of population values in successive years. And like so many of the images associated with chaos, this diagram is fractal, reflecting the universality of its structure: small parts of it turn out to resemble the whole.

Starting the Program

When you start Strange Attractors, you see the following screen:



The main image area of the screen shows the trails and airplanes of the strange attractor called the *Lorenz attractor*. The small white triangle is the *select cursor.*

On the lower left, you see three rectangular outlines called *stamps*. The program stores stamp-sized versions of your three most recent images in the stamp outlines. Just below the outlines is the *live-action stamp,* which shows the current attractor.

Exploring the Program

Try a few simple experiments with the strange attractors in this program before you learn how each feature works. You can learn more about the program's features later in this chapter.

Note: Use the status line when you first work with the program, to help you identify different types of attractors. To display the status line, press Alt+I. For details on the status line, see "Using the Status Line" on page 93.

Begin by displaying the other types of attractors.

1. With the Lorenz attractor displayed at the start of the program, press D to display the Yorke attractor.

   Notice the stamp image that appears in the first stamp outline, and the new live-action stamp that appears. For details on stamps, see "Using Stamps in the Program" on page 90.
2. Continue to press D to display the Hénon and Hénon Horseshoe attractors, and the Logistic Map, Logistic Pulse, and Logistic Hump attractors.
3. Finally, press D to display the Lorenz attractor again.

Now, try several experiments with color.

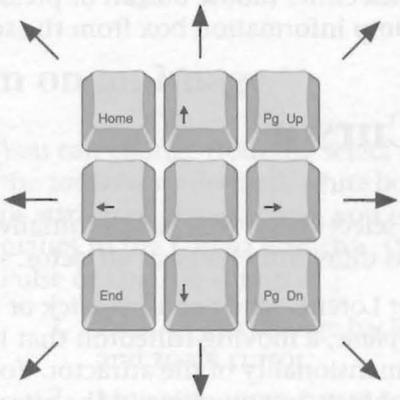
1. Display the Yorke attractor.
2. Press Alt+Y to cycle colors continuously in the attractor. Watch how different colors enhance the ripples of this attractor. You can also press Alt+R to redraw the image and watch the ripples start.
3. Press , (comma key) seven or eight times to slow down the attractor. Then press Alt+R to redraw the image. Watch the attractor slowly ripple into view as different colors play across the image.
4. Press Alt+Y two more times to turn the continuous color cycling feature off. Press Alt+D to return to the default palette. Press . (period key) seven or eight times to speed up the attractor again.

It's fun to listen to the attractors' "music" with the sound turned on.

1. With the Yorke attractor displayed, press Alt + V to turn the sound on. You hear an intense buzzing sound.
2. Press , (comma key) two or three times to slow down the attractor again. With the attractor slowed down, you can hear individual tones rise and fall.
3. Press . (period key) two or three times to speed up the attractor and the sound you're listening to.
4. Press D to display the next attractor.
5. Repeat steps 2–4 until you've listened to the sound of all the attractors in the program.
6. Press Alt + V to turn the sound off.

Moving Around in the Image Area

You can use the mouse or the keyboard to move the cursor around in the image area. To move around using the keyboard, make sure Num Lock is turned off and refer to the following illustration:



Numeric keypad keys and direction of cursor movement

Using Stamps in the Program

The program generates stamps for changes you make, such as displaying a new type of attractor or zooming in. (The program does not generate stamps when you make changes to the palette.) You can select one of the three images stored in the stamp outlines to display in the image area.

- Select the stamp icon you want by left-clicking on it. The new image displays in the image area.

Note: You can also press (Alt)+(1), (Alt)+(2), or (Alt)+(3) to select a stamp image to display in the image area (with stamp 1 being just above the live-action stamp and stamp 3 at the top).

You can get parameter information on a stamp image, too.

1. Right-click on a stamp icon to view the parameter information.

   If you're using a keyboard, you can press (Alt)+(4), (Alt)+(5), or (Alt)+(6) to display a stamp information box (with stamp 4 being just above the live-action stamp and stamp 6 at the top).
2. Click either mouse button or press any key to remove the stamp information box from the screen.

Using the Select Cursor

Using the select cursor (the cursor initially shown in the image area) works differently for each attractor.

- For Lorenz, when you left-click or press (Ins), you add an *airplane,* a moving trihedron that helps show the three-dimensionality of the attractor. You can have up to 80 airplanes at any one time in the attractor. When you right-click or press (Del), you remove an airplane. Once you've removed all airplanes, right-clicking or pressing (Del) removes one of the moving points. You cannot remove the last moving point.
- For Yorke and the two Hénon attractors, when you left-click or press (Ins), you add points to the attractor. When you right-click or press (Del), you remove points.

The points you add and remove for these three types of attractor can be subtle. For the Yorke attractor, the points you add are a new color from those already on screen. For the Hénon attractor, set the Flock Size option on the Tweaks menu to Small, and then add points. For the Hénon Horseshoe attractor, set the Flock Size option on the Tweaks menu to Medium, and then add points. Again, the points you add to Hénon and Hénon Horseshoe are a new color from those already on screen. When you remove points, you might help clarify the process by pressing Alt + R now and then to redraw the screen.

- For Logistic Map, when you left-click or press Ins, you change to Logistic Pulse, and, depending on the point you have selected in Logistic Map, you set the starting value of a parameter called *chaoticity*. (Right-clicking or pressing Del does not change anything for Logistic Map.)
- For Logistic Pulse and Logistic Hump, when you left-click or press Ins, you change the initial value of the point whose orbit is being tracked. When you right-click or press Del, you return to Logistic Map.

Zooming, Panning, and Centering

Zooming In on an Image

You can change from the select cursor (a small, white triangle) to the *zoom cursor* (a small, white box) and zoom in on different parts of attractors. For example, you might want a close-up of the airplanes in the Lorenz attractor. (You cannot zoom in on Logistic Pulse or Logistic Hump.)

1. Press E to change back and forth between select cursor and zoom cursor.
2. Move the zoom cursor to the desired location. (The cursor location becomes the center of the magnified image.)
3. Left-click or press Ins.
4. To zoom out, right-click or press Del.

For zooming in, you can also use a resizable zoom box that encloses any area you want to enlarge.

1. Press Alt + Z to display the zoom box cursor.
2. Move the zoom box to the location you want to zoom in on.
3. Hold down the Shift key and move the mouse to resize the zoom box.

   If you're using the keyboard, make the zoom box larger by pressing Shift + ↑ or Shift + ←. Make it smaller by pressing Shift + ↓ or Shift + →. (When you work with the Logistic Map attractor, you can automatically adjust the height and width of the zoom box independently. This allows you to fit the zoom box to areas of the attractor that are compressed-looking.)
4. Left-click or press Ins to zoom in on the specific area.
5. With the zoom box displayed, you can also zoom out by right-clicking or pressing Del. (After you zoom in or out with the zoom box, the cursor you were previously using reappears.)
6. If you change your mind when working with the zoom box, you can press Esc to return to the cursor you were previously using.

Panning an Image

When you pan an image, you can see the parts of it that are outside the edges of the image area.

1. Move the cursor to the location you want to pan to. (The cursor location becomes the center of the panned image.)
2. Press Alt + O to pan to the new location.

Centering an Image

Centering an attractor moves it to the center of the image area and unzooms it.

- Press Alt + C.

Examining an Attractor in Steps

You can examine an attractor's activity one step at a time. When you look at the attractor in steps, you get a better idea of its movement.

1. Press the spacebar to stop the attractor's activity.
2. Press the spacebar repeatedly to examine the activity one step at a time.
3. When you've finished viewing the activity in steps, press Shift and the spacebar to restart the regular activity.

Using the Status Line

You can turn on a status line that displays the name of the attractor currently showing and some parameter information for the attractor. (The status line for Hénon Horseshoe displays the attractor's name but no parameter information.)

The following table summarizes the parameter information you can find in each attractor's status line.

Parameter	Lor.	Yorke	Hénon	Map	Pulse	Hump
Axis (View)	x					
Accuracy	x					
Topology (World Shape)		x				
Chaoticity		x	x		x	x
Humpspot				x	x	x

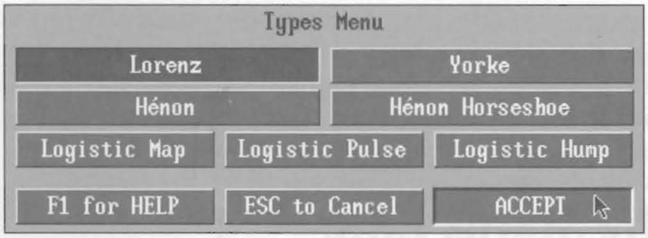
To toggle the status line on and off

- Press Alt+I.

Selecting an Attractor

Keyboard shortcut: D

To select an attractor, left-click the F3 Types button at the upper-left corner of the screen, or press F3. The Types menu appears:



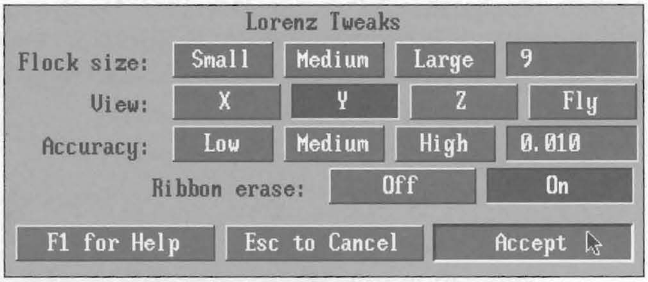
- Select the button showing the name of the attractor you want to see.

Using the Lorenz Attractor

The *Lorenz* attractor is a set of points moving in three-dimensional space. The attractor is based on three equations for convection discovered by meteorologist Edward Lorenz. The Lorenz attractor's moving point shows the chaotic rotation of a fluid. In this program's Lorenz attractor, the moving points are shown by dots or by "airplanes"—trihedrons that twist and turn to follow the three-dimensional contortions of the attractor's curving trajectory. The two "ears" of the attractor correspond to the two directions a cylinder of fluid might rotate in.

Tweaking Lorenz

With the Lorenz attractor displayed, left-click the F4 Tweak button at the upper-left corner of the screen, or press F4. The Lorenz Tweaks menu appears:



Lorenz Tweaks

Flock size: Small | Medium | Large | 9

View: X | Y | Z | Fly

Accuracy: Low | Medium | High | 0.010

Ribbon erase: Off | On

F1 for Help | Esc to Cancel | Accept

Use the options on this menu to change parameters for the Lorenz attractor.

Changing Flock Size

Keyboard shortcut: V to cycle through small, medium, and large flock sizes

When you change the *flock size* in the Lorenz attractor, the program removes all of the airplanes, and it changes the number of trails shown in the attractor so you can focus on them. (Working with the select cursor, you can add a total of 80 airplanes back in when you left-click or press Ins.) You can change the flock size to a small, medium, or large number of trails, or you can enter the number of trails in the entry field at the right side of the menu. The flock size settings on the buttons correspond to 1 (Small), 27 (Medium), and 64 (Large). The default setting in the entry field is 9.

- Select a size button (Small, Medium, Large), or enter a value between **1** and **64** and press ↵.

Changing View

Keyboard shortcut: X to select x-axis; Y to select y-axis; Z to select z-axis; W to select fly view

Selecting a *view* sets the viewing orientation showing the axis of the Lorenz attractor's spirals. You can view from the x-, y-, or z-axis. For example, the x view projects the attractor onto the yz plane. You can also select fly view, in which the program moves a set of axes to the location of one of the airplanes and projects an image of the attractor onto the plane of that airplane's tail and wings. If you've been adding airplanes, it uses the newest one. The default setting is the y-axis.

- Select a view button (X, Y, Z, or Fly).

Suggestion: When you select fly view, you can observe more easily with just one airplane in the attractor. Right-click or press Del to delete airplanes until you have just one airplane left.

Changing Accuracy

Keyboard shortcut: Shift + U to increase; U to decrease

CHAOS updates the attractor step-by-step in the program's own simulation time. Each step takes the same amount of real time to calculate. *Accuracy* provides an approximate range of steps the program performs per second of simulation time. When you change accuracy, you affect how slowly or rapidly the attractor moves, and how smooth or jagged its trail looks.

You can select a low, medium, or high accuracy, or you can type an accuracy value in the entry field at the right side of the menu. You can enter a setting in integers. The default setting in the entry field is 100.

- Select an accuracy button (Low, Medium, High), or enter a value between **10** and **10000** and press ↵.

Suggestion: If you have a slower computer, you might prefer to set a low accuracy until you've found an area you wish to study in detail.

Setting Ribbon Erase

Keyboard shortcut: T

The Ribbon Erase parameter lets you change how the attractor's trails are painted on screen. With Ribbon Erase off, trails are shown as continuous lines. With Ribbon Erase on (the default), trails appear as constantly moving, fixed-length lines. Using this parameter helps you see the attractor's trajectory in phase space.

Note: You can change the length of the displayed trails with the Trace Length option on the Options menu. See "Trace Length" on page 114 for details.

- Select the On or Off button.

Turning Trails Off and On

You can turn airplane trails off and on for the Lorenz attractor. This helps you see the twisting movement of the trihedrons.

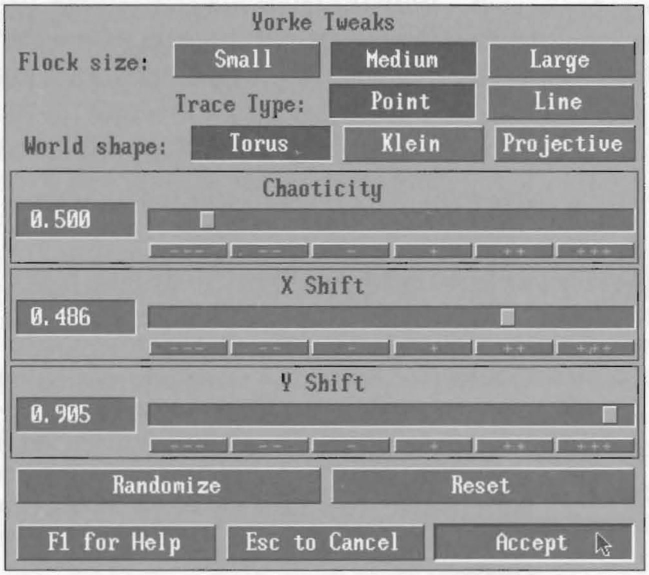
1. Set the Flock Size option on the Tweak menu to Small.
2. With the select cursor displayed, left-click or press Ins and insert as many airplanes as you want.
3. Press S to turn trails off. (Press Alt + R to clean up old trails.) Watch the trihedrons' movement.
4. When you're ready to work with trails on, press S again and set flock size the way you want.

Using the Yorke Attractor

The *Yorke* attractor is a set of points moving in a two-dimensional plane. The points for this attractor—and for the Hénon and Hénon Horseshoe attractors—are like fluorescent fleas that hop around and mark the attractor's outline. Discovered by mathematician James Yorke, this attractor involves the interplay of different rhythms (like planetary orbits) that produce a special kind of chaos.

Tweaking Yorke

With the Yorke attractor displayed, left-click the F4 Tweak button at the upper-left corner of the screen, or press F4. The Yorke Tweaks menu appears:



Yorke Tweaks

Chaoticity: 0.500

X Shift: 0.486

Y Shift: 0.905

Use the options on this menu to change parameters for the Yorke attractor.

Changing Flock Size

Keyboard shortcut: V to cycle through small, medium, and large flock sizes

Setting a flock size changes the number of points moving around on the strange attractor. When you set a lower flock size for Yorke, you can see the gnarled patterns being formed by the points. You can change the flock size to a small, medium, or large number of points. The default setting is Medium.

- Select a size button (Small, Medium, Large).

Setting Trace Type

Keyboard shortcut: T to toggle between points and lines

When you select a *trace type,* you change how the program releases points for the strange attractor. The program can release points as individual dots (the default), or as lines. When you set trace type to Line, the program draws lines connecting successive point positions, erasing the old lines according to the length of the trail. Points that wrap around the screen in Yorke are not computed.

- Select the Point or Line button.

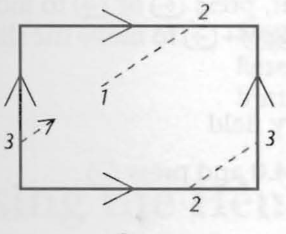
Suggestion: Set trace type to Line, and redraw the screen with Alt + R. Then press A or Shift + A several times to see the dance of the attractor. It's also interesting to do this experiment with the sound turned on (press Alt + V).

Changing World Shape

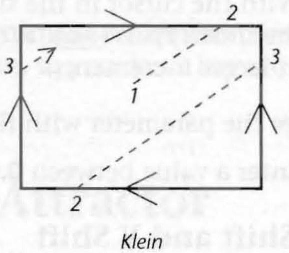
Keyboard shortcut: W to cycle through Torus, Klein, and Projective world shapes

In the Yorke attractor, points *wrap,* or go off screen and come back on at different locations. The three different world shapes of the Yorke attractor relate to how points on the screen wrap. The default setting is Torus.

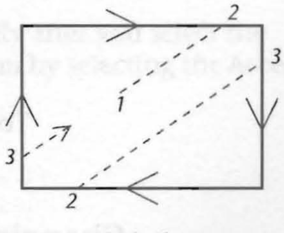
The following illustration shows how points on the screen wrap for each world shape:



Torus



Klein



Projective

- Select a world shape button (Torus, Klein, Projective).

Suggestion: After you change to a new world shape, you might want to randomize repeatedly to see how different calculations change the world shape. Press N repeatedly to randomize the attractor. You might also want to try different values of chaoticity after you randomize. See the next section for several keyboard shortcuts that allow you to adjust chaoticity.

Changing Chaoticity

Keyboard shortcut: Shift + A increases by .1; A decreases by .1; Shift + B increases by .01; B decreases by .01; Shift + C increases by .001; C decreases by .001

When you set the *chaoticity* parameter, you change internal calculations that control how messy or formless the attractor looks. Changing chaoticity weakens or strengthens the feedback loop of calculations, in which equations are repeated again and again. For higher chaoticity, the result is more messy looking attractors; for lower chaoticity, less messy looking attractors. The default setting for chaoticity in Yorke is 0.500.

To change the parameter with the mouse on the slider bar

- Hold down the left mouse button and drag the box in the slider to the setting you want. The setting in the entry field at the left changes as you move the box.

  or

- Left-click the plus or minus buttons below the slider bar. The greater the number of plus symbols on the button, the larger the increment.

To change the parameter with the keyboard for the slider bar

- With the cursor in the slider bar, press ← or → to move the slider. Press Shift + ← or Shift + → to move the slider in larger increments.

To change the parameter with the entry field

- Enter a value between **0.0** and **4.0** and press ↵.

Changing X Shift and Y Shift

For each setting of x shift and y shift, there's a different sequence of Yorke attractors. If you set chaoticity to 0 with x and y shift parameters at their defaults, the program produces diagonal patterns of points. The x and y shift parameters control the direction of the diagonal patterns. When you increase chaoticity, the patterns become more ridged-looking. Finally, as you continue to increase chaoticity, the patterns begin to break up into chaos. The default setting for x shift is 0.486, and for y shift, 0.905.

To change x shift and y shift parameters, follow the instructions for using the slider bar in the previous section, "Changing Chaoticity."

To change x shift and y shift with the entry fields

- Enter a value between –1 and 1 and press ↵.

Randomizing Yorke

Keyboard shortcut: N

You can randomize two Yorke parameters: x and y shift. The first five times you randomize, you get the same preset values. These preset values were discovered by James Yorke and published in one of his papers.

- Select the Randomize button.

Note: Randomization happens immediately after you select the Randomize button, so you don't have to confirm by selecting the Accept button. If you want to check what values x shift and y shift changed to, check the Yorke Tweaks menu.

Resetting Yorke

Keyboard shortcut: Shift + N

You can reset the original parameters for chaoticity and x and y shift.

- Select the Reset button.

Note: Resetting also happens immediately after you select the Reset button, so you don't have to confirm by selecting the Accept button.

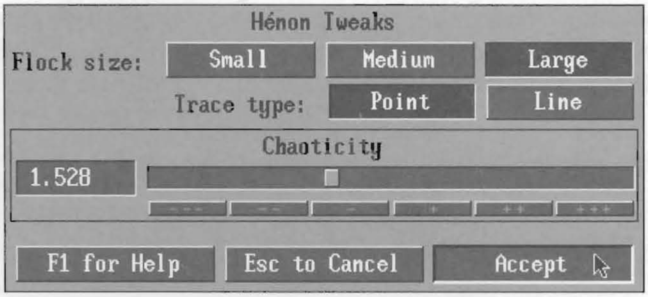
Using the Hénon Attractor

Keyboard shortcut: N toggles two Hénon attractor types (only when Hénon or Hénon Horseshoe is displayed)

The *Hénon* attractor is a set of points moving in a two-dimensional plane. Modeled by French astronomer Michel Hénon, the Hénon attractor shows stellar orbits around a galactic center. The points outline a *torus* (a doughnut shape), with separate toruses around the central torus. Unstable orbits scatter points around randomly. The final image is one of disorder surrounding order, of islands around other islands. With magnification, other islands on smaller and smaller scales appear.

Tweaking Hénon

With the Hénon attractor displayed, left-click the F4 Tweak button at the upper-left corner of the screen, or press F4. The Hénon Tweaks menu for the Hénon attractor appears:



Hénon Tweaks

Chaoticity: 1.528

Use the options on this menu to change parameters for the Hénon attractor.

Changing Flock Size

Keyboard shortcut: V to cycle through small, medium, and large flock sizes

Setting a flock size changes the number of points moving around on the strange attractor. When you change to a lower flock size, the program removes the islands and leaves just one or several rings of points. (Working with the select cursor, you can add points back in when you left-click or press Ins.) You can change the flock size to a small, medium, or large number of points. These correspond to flock sizes of 1, 4, and 64, respectively.) The default setting is Large.

- Select a size button (Small, Medium, Large).

Setting Trace Type

Keyboard shortcut: T to toggle between points and lines

For a description of this parameter and instructions on how to change it, see "Setting Trace Type" on page 99.

Note: Setting a line trace type works best for Hénon with a medium flock size.

Changing Chaoticity

Keyboard shortcut: (Shift)+(A) increases by .1; (A) decreases by .1; (Shift)+(B) increases by .01; (B) decreases by .01; (Shift)+(C) increases by .001; (C) decreases by .001

In Hénon, each setting of chaoticity gives a pattern in which points either move periodically or race off screen. A point that goes off screen is thrown back on screen in search of a periodic orbit. As the points hunt for periodic orbits, they draw a messy, chaotic outer region, and an orderly inner region. Sometimes it is necessary to zoom out to see both regions. For instructions on how to change this parameter, see "Changing Chaoticity" on page 100. The default setting for chaoticity in Hénon is 1.528.

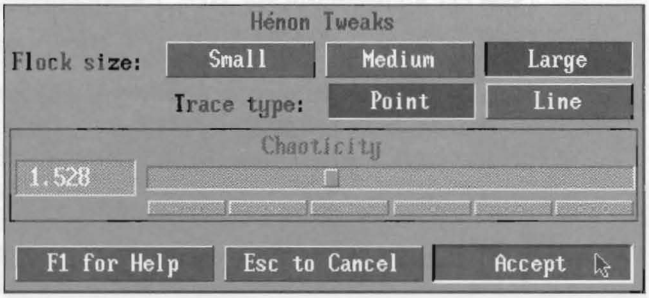
Using the Hénon Horseshoe Attractor

Keyboard shortcut: (N) toggles two Hénon attractor types (only when Hénon or Hénon Horseshoe is displayed)

The *Hénon Horseshoe* attractor is a set of points moving in a two-dimensional plane. Hénon Horseshoe shows an attractor where phase space is repeatedly stretched and folded into many thin layers. The points outline a crescent shape consisting of many lines of points.

Tweaking Hénon Horseshoe

With Hénon Horseshoe displayed, left-click the F4 Tweak button at the upper-left corner of the screen, or press F4. The Hénon Tweaks menu for Hénon Horseshoe appears:



Hénon Tweaks

Flock size: Small Medium Large

Trace type: Point Line

Chaoticity

1.528

Note: CHAOS dims the chaoticity parameter on this menu, which means that this parameter is not available for Hénon Horseshoe.

Use the options on this menu to change parameters for the Hénon Horseshoe attractor.

Changing Flock Size

Keyboard shortcut: V to cycle through small, medium, and large flock size

Setting a flock size changes the number of points moving around on the strange attractor. When you change to a lower flock size, the program removes points and leaves just one or several lines of points. (Working with the select cursor, you can add points back in when you left-click or press Ins.) You can change the flock size to a small, medium, or large number of points. These correspond to flock sizes of 1, 2 and 16, respectively. The default setting is Large.

- Select a size button (Small, Medium, Large).

Setting Trace Type

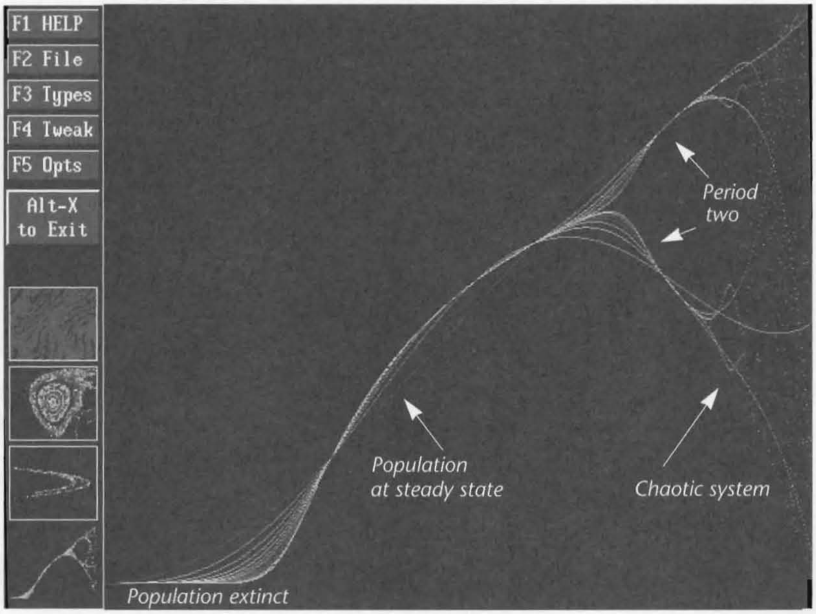
Keyboard shortcut: T to toggle between points and lines

For a description of this parameter and instructions on how to change it, see "Setting Trace Type" on page 99.

Using the Logistic Map Attractor

The *Logistic Map* attractor demonstrates a route to chaos by *bifurcation,* or splitting into two. Developed by biologist Robert May, the attractor shows how changes in one parameter for an equation relating to population behavior affect the behavior of the entire system. Values of this parameter are shown from left to right, with the final population plotted on the vertical axis.

When you look at the Logistic Map, you see the following areas of population behavior in the equation:



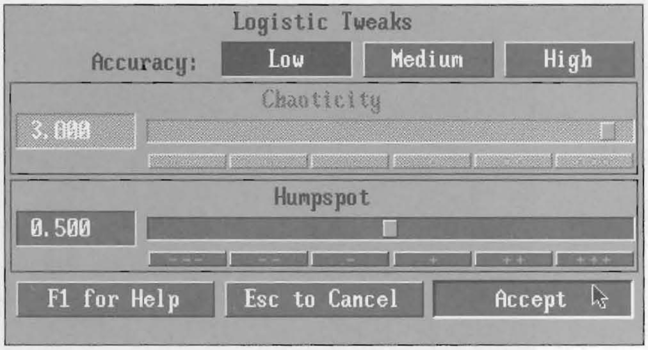
With a low parameter set in the equation, the population becomes extinct. Increasing the parameter makes the equilibrium level of the population rise. At a certain point (known as *period two)* when the population rises, the equilibrium splits in two, and the population level begins to alternate. These bifurcations start coming more rapidly—and suddenly, the system turns chaotic, with the population going to an infinite number of values.

Although the chaotic region seems to be nothing but disorder, you can find order there. Period doubling appears in different parts of the chaotic area. When you magnify certain parts of the seemingly random region, you see that they resemble the whole attractor.

Note: For the Logistic Map attractor, the screen paints repeatedly, with higher and higher precision. After the first few screen paints, Logistic Map leaves all points on screen, even though its calculations get more accurate. After letting it run awhile, you can press Alt+R to clean up the image.

Tweaking Logistic Map

With the Logistic Map attractor displayed, left-click the F4 Tweak button at the upper-left corner of the screen, or press F4. The Logistic Tweaks menu for Logistic Map appears:



Chaoticity: 3.000

Humpspot: 0.500

Note: CHAOS dims the Chaoticity parameter on this menu, which means that this parameter is not available for Logistic Map.

Use the options on this menu to change parameters for the Logistic Map attractor.

Changing Accuracy

Keyboard shortcut: Shift+U to increase; U to decrease

When you change *accuracy,* you control how many separate point positions the program computes for each vertical cross-section through the map. You can select a low, medium, or high accuracy. The default setting is Low.

- Select an accuracy button (Low, Medium, High).

Suggestion: You can see much more detail when you zoom in on the Logistic Map attractor if you set accuracy to high.

Changing Humpspot

Keyboard shortcut: (Shift)+(O) to increase; (O) to decrease

The Humpspot parameter changes the shape of the curve used to compute the function underlying the Logistic imagery. The humpspot is the horizontal location of the high point of the curve shown in the Logistic Hump attractor. The default Humpspot setting is 0.500.

To change the parameter with the mouse on the slider bar

- Hold down the left mouse button and drag the box in the slider to the setting you want. The setting in the entry field at the left changes as you move the box.

  or

- Left-click the plus or minus buttons below the slider bar. The greater the number of plus symbols on the button, the larger the increment.

To change the parameter with the keyboard for the slider bar

- With the cursor in the slider bar, press (←) or (→) to move the slider. Press (Shift)+(←) or (Shift)+(→) to move the slider in larger increments.

To change the parameter with the entry field

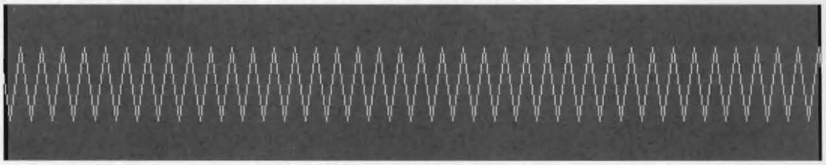
- Enter a value between **0.1** and **0.9** and press (↵).

Using Logistic Pulse

Keyboard shortcut: (T) (only with Logistic Map or Logistic Hump displayed)

Logistic Pulse is a pulse line linked to Logistic Map. Each vertical cross-section of the Logistic Map corresponds to a different Logistic Pulse. When you left-click or press (Ins) in a region of Logistic Map, Logistic Pulse demonstrates what kind of periodicity there is in the region (period two, period four, and so on), or whether there is chaos there. Then you can right-click or press (Del) to return to Logistic Map.

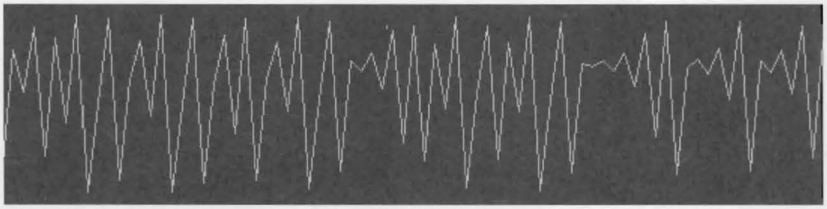
For example, left-click or press Ins in the region to the right of the first bifurcation (the region labeled "Period Two" in the screen illustration for Logistic Map). Here's part of what you see on screen:



Period two

To find out what periodicity you are looking at, count the points on the sawtooth curve until the pattern recurs. In the previous illustration, you can count just two points before the same pattern repeats, so you have period two.

When you left-click or press Ins in the chaotic region of Logistic Map, here's part of what you see on screen:

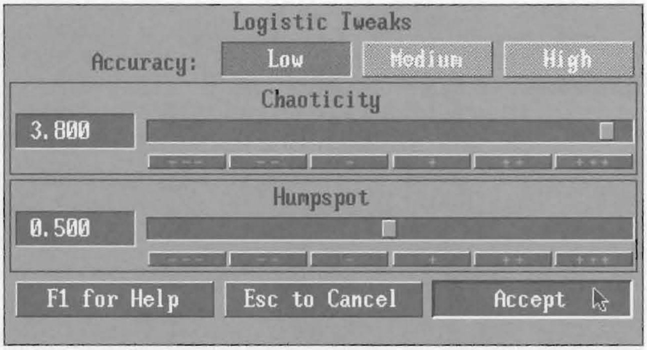


Chaos

You cannot find a recurring pattern of points to count in the illustration of this pulse, so you know you are looking at a chaotic region.

Tweaking Logistic Pulse

With Logistic Pulse displayed, left-click the F4 Tweak button at the upper-left corner of the screen, or press F4. The Logistic Tweaks menu for Logistic Pulse appears:



Chaoticity: 3.800

Humpspot: 0.500

Note: CHAOS dims the Accuracy parameter on this menu, which means that this parameter is not available for Logistic Pulse.

Use the options on this menu to change parameters for Logistic Pulse.

Changing Chaoticity

Keyboard shortcut: Shift+A increases by .1; A decreases by .1; Shift+B increases by .01; B decreases by .01; Shift+C increases by .001; C decreases by .001

The Chaoticity parameter determines which vertical cross-section of Logistic Map the Pulse is based on. Setting chaoticity values between 3.5 and 4.0 shows the greatest variety. For instructions on how to change this parameter, see "Changing Chaoticity" on page 100. The default setting for chaoticity in Logistic Pulse is 3.800.

- Enter a value between **0.001** and **3.999** and press ↵.

Changing Humpspot

Keyboard shortcut: Shift+O to increase; O to decrease

For a description of this parameter and instructions on how to change it using the slider bar, see "Changing Humpspot" on page 107.

To change the parameter using the entry field

- Enter a value between **0.1** and **0.9** and press ↵.

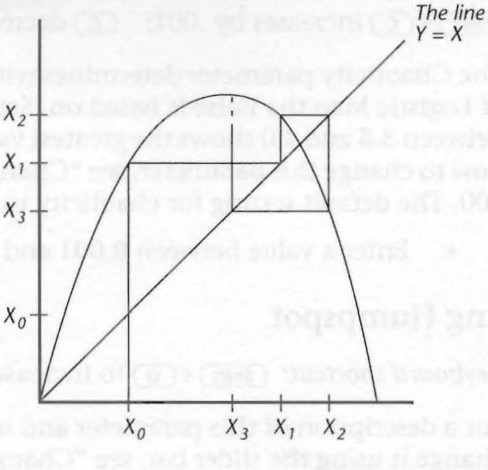
Using the Logistic Hump Attractor

Keyboard shortcut: T (only with Logistic Pulse displayed)

The *Logistic Hump* attractor shows a parabolic curve with lines chasing around it in a spiderweb pattern. This represents the calculations underlying the Logistic Pulse. In trying to understand the boundary region between order and chaos, mathematician Mitchell J. Feigenbaum discovered the principle of *universality*. Here, universality means that the same pattern of period-doubling can be found on differently shaped curves. Feigenbaum saw that period-doublings mark a transition near the region between order and chaos.

The Logistic Hump is a way to visualize one year's population and the next year's. Instead of a steady, limitless growth for the population, or a parabola where the population goes back down, Feigenbaum's principle of universality proposed a critical "hump," which depended on steepness or the degree of nonlinearity in the equation.

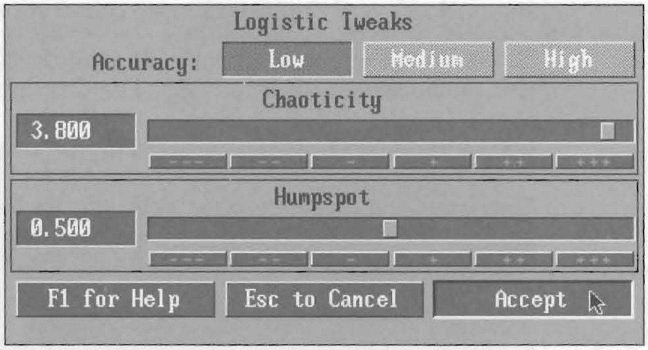
Here's an illustration of how this works:



Suggestion: As you work with the Logistic Hump, you should keep pressing Alt+R to redraw the image. Examining this attractor in single steps helps you see its patterns, too. For details, see "Examining an Attractor in Steps" on page 93.

Tweaking Logistic Hump

With Logistic Hump displayed, left-click the F4 Tweak button at the upper-left corner of the screen, or press F4. The Logistic Tweaks menu for Logistic Hump appears:



Logistic Tweaks

Accuracy: Low Medium High

Chaoticity

3.800

--- -- - + ++ +++

Humpspot

0.500

--- -- - + ++ +++

F1 for Help | Esc to Cancel | Accept

Note: CHAOS dims the Accuracy parameter on this menu, which means that this parameter is not available for Logistic Hump.

Use the options on this menu to change parameters for Logistic Hump.

Changing Chaoticity

Keyboard shortcut: Shift+A increases by .1; A decreases by .1; Shift+B increases by .01; B decreases by .01; Shift+C increases by .001; C decreases by .001

The Chaoticity parameter determines which vertical cross-section of Logistic Map the Hump is based on. Setting chaoticity values between 3.5 and 4.0 shows the greatest variety. For instructions on how to change this parameter, see "Changing Chaoticity" on page 100. The default setting for chaoticity in Logistic Hump is 3.800.

- Enter a value between **0.001** and **3.999** and press ↵.

Changing Humpspot

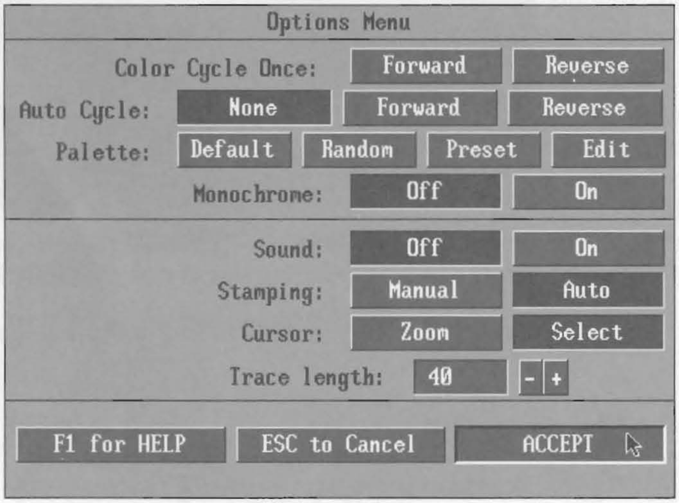
Keyboard shortcut: Shift + O to increase; O to decrease

For a description of this parameter and instructions on how to change it, see "Changing Humpspot" on page 107.

Using the Options Menu

With the Options menu, you can turn on color, sound, and stamping options, and you can change cursor mode and set trace length.

To select options, left-click the F5 Opts button at the upper-left corner of the screen, or press F5. The Options menu appears:



Color Options

The following color options are common to five of the CHAOS programs. For details on these options, see "Using Color Options" on page 214.

- Color Cycle Once
- Auto Cycle
- Palette (Default, Random, Preset, Edit)
- Monochrome

Cursor Mode

Keyboard shortcut: E

When you work with strange attractors in this program, you can use either the select cursor or the zoom cursor. For details on working with the select cursor, see "Using the Select Cursor" on page 90. To find out how to use the zoom cursor, see "Zooming In on an Image" on page 91. The default setting is select cursor.

- Select the Zoom or Select button to work with the cursor you want.

Suggestion: If you're unsure where you are after zooming, you can always re-center the image by pressing Alt + C.

Stamping

Keyboard shortcut: Alt + T

You can work with automatic stamping or manual stamping. Automatic stamping saves a stamp for every change you make. Manual stamping saves a stamp only when you press the keyboard shortcut shown here. The default is automatic stamping.

- Select the Automatic or Manual button to save stamps either automatically or only when you request it.

Sound

Keyboard shortcut: Alt + V

When you turn the sound on in Strange Attractors, you hear multiple tones that follow the movement of points in each attractor. The default is sound off.

- Select the On or Off button to turn sound on or off.

Suggestion: Try slowing down an attractor when you turn the sound on. Press , (comma key) to slow down the sound and listen to individual points being released. Press . (period key) to speed up the activity again.

Trace Length

Keyboard shortcut: Shift + L to increase; L to decrease

Changing trace length affects the length of the trails in Lorenz, Yorke, Hénon, and Hénon Horseshoe. Changing trace length does not affect any of the Logistic attractors. You can change the trace length for Lorenz only if you have set the Ribbon Erase option on the Lorenz Tweaks menu to On. You can alter the trace length for Yorke, Hénon, and Hénon Horseshoe only if you have selected the Line option on each Tweaks menu.

The default settings are 40 for the Lorenz attractor, and 3 for the Yorke, Hénon, and Hénon Horseshoe attractors. You can enter a trace length between 16 and 128 for the Lorenz attractor; and between 4 and 128 for the Yorke, Hénon, and Hénon Horseshoe attractors.

- Enter the new value for trace length and press ↵. You can also increase or decrease trace length by using the + or − buttons next to the Trace Length option.

Saving and Loading Files

Use the options on the File menu to save and reload parameter files and *.gif* files. For details on saving and reloading files, see “Saving and Loading Files” on page 209.

The Mathematics of the Program

This program shows strange attractors in several kinds of dynamical systems.

A dynamical system consists of a space plus a “dynamic” that tells how particles in this space are supposed to move. The underlying space can be a 1D line, a 2D surface, a 3D space, or a space of higher dimensions.

Examples of higher-dimensional dynamical systems occur when physicists talk about systems with many independent variables. If, for instance, we were to be interested in a system of three moving particles, each with three space coordinates and three velocity components, then the complete state of the system would involve specifying six numbers per particle for a total of eighteen numbers in all. The state of the system would correspond to a location in an eighteen-dimensional "phase space."

Higher-dimensional spaces are hard to visualize, so in the program, the Lorenz attractor is part of a three-dimensional dynamical system, the Yorke and Hénon attractors are in two-dimensional dynamical systems, and the Logistic map is based on a one-dimensional dynamical system.

The "dynamic" that fills the space of a dynamical system is a systematic method for moving from point to point. This means that for each point P of the space, the dynamic picks a unique next point, which is called new(P). In formal terms, we can call the space S and say that the dynamic is a one-to-one function that maps S into S.

If we pick an arbitrary point P as a starting point, we can look at the point's orbit, which consists of the points P, new(P), new(new(P)), and so on. In some cases it's useful to draw lines connecting the successive points of the orbit, but keep in mind that it may often happen that the successive points of the orbit are not close to one another at all.

We can watch the action of the dynamic by releasing a test particle into the dynamical system at a point P. Now if it turns out that the orbits of many test points tend to cycle around a certain region of space, then this region is called an *attractor*, and if the attractor is complicated in an interesting way, it is called a *strange attractor*.

The rules by which the dynamic moves a particle from point P to the point new(P) are sometimes given as equations that give the coordinates of new(P) in terms of the coordinates of P. This is the way in which the Yorke, Hénon and Logistic attractors are described.

The description of the dynamic that yields the Lorenz attractor is a bit more complicated. Here, instead of specifying the coordinates of new(P) in terms of P, we specify the velocity in terms of P. If we imagine a time interval dt elapsing between the observation of P and new(P), we can think of the Lorenz equations as specifying a vector velocity (new(P) – P)/dt in terms of P.

A vector velocity actually involves three speeds: the speed in the x-direction, the speed in the y-direction, and the speed in the z-direction. If the velocity is taken over a very short time interval dt, we are effectively looking at three derivatives, dx/dt, dy/dt, and dz/dt. The Lorenz equations are so-called differential equations, which affect the motion of a point in 3D space like an odd force field. The equations are

$$dx/dt = 10*(y - x)$$
$$dy/dt = (28*x) - y - (x*z)$$
$$dz/dt = -(8/3)*z + (x*y)$$

Using these equations on a particle at any point P(x,y,z) in the space of the Lorenz attractor, we find the direction (dx/dt, dy/dt, dz/dt) which the particle should move in next. Depending on whether our Accuracy parameter is set to High or to Low, the particle then takes a small or a large step in the direction given by the equations. To make the simulation even more accurate, we actually take a weighted average of the velocity predicted for the particle at several nearby points. This trick is known as the Runge-Kutte method, and is available as an option in our Magnets and Pendulum program as well.

The exact values of the three parameters 10, 28, and –8/3 were found by Lorenz to yield the nicest image. We could have allowed the user to try tweaking these parameters to different values—but none of the different values look as good. Instead, the user can tweak the simulation by changing the accuracy, the direction from which the attractor is viewed, the number of particles moving around the attractor, and the types of trails the particles leave.

The Lorenz equations actually describe the motion of a ring of hot and cool air around an atmospheric convection cell. The x parameter represents the clockwise or counterclockwise motion of the air, and the y and z parameters measure the distribution of temperatures around the ring. The moving points correspond to evolving states of system; when a point switches from one "ear" of the attractor to the other, this corresponds to a rotating ring changing its direction of rotation.

The Hénon, Yorke, and Logistic attractors are conceptually much simpler. Instead of involving differential equations, these systems give the coordinates of new(P) directly in terms of P. In each of these systems, we single out one of the parameters for the user to tweak, and call the parameter the chaoticity.

The Hénon attractors lie in two-dimensional space. They are generated by a dynamic that maps a point (x,y) into the point (newx,newy), where newx and newy are specified by a chaoticity parameter A, and by the equations:

$$newx = x*\cos(A) - (y - x^2)*\sin(A)$$
$$newy = x*\sin(A) - (y - x^2)*\cos(A)$$

On our Hénon Tweaks menu, the user can move a parameter B between 0 and 4. The parameter A used in the Hénon equations is actually B*(π/4), so that as the user moves B from 0 to 4, A moves from 0 to π, covering half of the possible combinations of sine and cosine. (Keep in mind that in the so-called radian measure that scientists use, π/2 is 90 degrees, and 2*π is 360 degrees.) If we were to let A move through the range π to 2*π, we'd get rotated versions of the same images that occur between 0 and π.

As a special case, our program also shows the so-called Hénon Horseshoe map, which is given by the equations:

$$new\ x = 1 + y - 1.4*x^2$$
$$new\ y = 0.3*x$$

The Yorke maps are based on two particular functions TRIGX(x,y) and TRIGY(x,y), on two constants x shift and y shift, and on a chaoticity parameter A. The equations are

newx = oldx + x shift + A*(TRIGX(oldx, oldy)) WRAP 1
newy = oldy + y shift + A*(TRIGY(oldx, oldy)) WRAP 1

TRIGX and TRIGY are each a fixed linear combination of four sine functions of linear combinations of x and y. WRAP 1 means that if a value is larger than 1 or less than 0, Yorke wraps it back around to a value between 0 and 1. Thus, for instance, 1.13 becomes 0.13, –0.33 becomes (1 – 0.33) = 0.67, 2.124 becomes 0.124, and so on. This "torus wrap" is analogous to the "modulo" operator for integers. There are actually two other ways of wrapping that we allow: a "Klein bottle" wrap (in which the vertical axis gets a half twist) and a "projective plane" wrap (both axes get half twists). In a Klein bottle wrap, for instance, the point (x,1.3) gets wrapped to the point (1 – x,0.3), provided x is between 0 and 1.

On our Yorke Tweaks menu, the user can move a parameter B between 0 and 4. The parameter A used in the Yorke equations is actually B / (2*π), so that as the user moves B from 0 to 4, A moves from 0 to 2/π.

The TRIGX and TRIGY maps are given in terms of no less than sixteen parameters!

$$
\begin{aligned}
A110 &= -0.26813663648754\\
B110 &= 0.98546084298505\\
A210 &= 0.08818611671542\\
B210 &= 0.99030722865609\\
A101 &= -0.91067559396390\\
B101 &= 0.50446045609351\\
A201 &= -0.56502889980448\\
B201 &= 0.33630697012268\\
A111 &= 0.31172026382793\\
B111 &= 0.94707472523078\\
A211 &= 0.16299548727086\\
B211 &= 0.29804921230971\\
A11m1 &= -0.04003977835470\\
B11m1 &= 0.23350105508507\\
A21m1 &= -0.80398881978155\\
B21m1 &= 0.15506467277737
\end{aligned}
$$

$$
\begin{aligned}
TRIGX(x,y) = {} & A110 \ast \sin(2 \ast \pi \ast (x + B110)) +\\
& A101 \ast \sin(2 \ast \pi \ast (y + B101)) +\\
& A111 \ast \sin(2 \ast \pi \ast (x + y + B111)) +\\
& A11m1 \ast \sin(2 \ast \pi \ast (x - y + B11m1))
\end{aligned}
$$

$$
\begin{aligned}
TRIGY(x,y) = {} & A210 \ast \sin(2 \ast \pi \ast (x + B210)) +\\
& A201 \ast \sin(2 \ast \pi \ast (y + B201)) +\\
& A211 \ast \sin(2 \ast \pi \ast (x + y + B211)) +\\
& A21m1 \ast \sin(2 \ast \pi \ast (x - y + B21m1))
\end{aligned}
$$

Where on earth did we come up with such a gnarly definition of a function? We copied it out of a paper by Yorke! Where did he get it? By a process of informed trial and error. And what about the x shift and y shift parameters? These are what you change when you press the $\boxed{\text{N}}$ key. The first five times you press $\boxed{\text{N}}$, you get preset values of x shift and y shift which we took from the same paper by Yorke. But from then on these values are taken to be random real numbers between 0 and 1. The remarkable thing is that no matter what values of x shift and y shift you end up with, you can tweak the Chaoticity to a value which will make the attractor look good.

The Logistic map tracks a point in 1D space acting under a dynamic that takes x to newx according to the equation:

$$\text{new } x = A \ast x \ast (1 - x)$$

In Logistic Pulse and Logistic Hump, the chaoticity parameter A is adjusted directly on the Logistic Tweaks menu to lie between 0 and 4.

In the Logistic Map picture, all the possible values of A from 0 to 4 are used, with this parameter running along the horizontal axis from left to right. The successive x values are plotted vertically, with a different color used for each successive x value.

In the Logistic Pulse pictures, we show the successive values of x as points on a sawtooth curve. Left-clicking the mouse in this simulation selects different start values for x.

In the Logistic Hump pictures, we show a graphic image of how the successive values of x are computed. The parabola is a graph of the equation $y = A*x*(1 - x)$, where x is the horizontal axis. The diagonal line is a graph of the equation $y = x$. Left-clicking the mouse chooses a new horizontal starting position for x. Moving up to the parabola sets y equal to the newx values $A*x*(1 - x)$. Going across from the parabola to the diagonal line finds the value of x that is equal to this newx y value. Moving vertically from that line point to the parabola again gets the new newx value, and so on. (See the illustration in the section "Using the Logistic Hump Attractor" on page 110.)

As it turns out, the same attractor behavior occurs for other simply humped graphs besides the parabola $A*x*(1 - x)$. In particular, if n is any number, then the equation:

$newx = [\, r*x^n(1 - x^n)\,]^{(1/n)}$

yields the same behavior.

Chapter 5
The Chaos Game

Barnsley and his co-workers now embarked on an out-of-control program of producing pictures, cabbages and molds and mud . . . As points flash across the computer screen, no one can guess where the next one will appear; that depends on the flip of the machine's internal coin. Yet somehow the flow of light always remains within the bounds necessary to carve a shape in phosphorous. The role of chance is an illusion . . . (Chaos: Making a New Science, *page 237.)*

Entranced by the mathematical beauty of Mitchell Feigenbaum's attractor and Benoit Mandelbrot's fractals, an Oxford-educated mathematician, Michael Barnsley, set out to create one of the most intensely practical applications of the new science. He called it the Chaos Game. It is a way of taking simple rules, adding a bit of randomness to the stew, and producing recognizable images.

Imagine that the computer screen is a map of Colorado. You start anyplace—Denver, say. You are going to hop from Denver to a new place, according to a certain rule. Then you are going to apply the rule again, and again, and again, leaving a dot on the screen each time, to trace your progress.

Only simple rules are allowed. For example, the rule might be, move directly north, half the distance to the border (or the edge of the screen). Suppose you applied that rule not just to a single point but to a region—say, the entire city of Denver. You would get a new region, relocated halfway to Wyoming, the same size as Denver in the east-west direction but only half the size in the north-south direction. Denver will have been moved and squashed. In fact, the way to visualize the rule is to think about what it does to any region. In this case, it moves the region north and squashes it top and bottom. Other rules might stretch a region east and west; or rotate a region clockwise 30 degrees; or "flip" a region into its mirror image; or "shear" a region into a parallelogram, by pulling the top to the west and the bottom to the east. In fact those are the only rules allowed: shrink or stretch, rotate clockwise or counterclockwise, flip or shear.

To play the Chaos Game, you take more than one rule, perhaps just two or three, perhaps a half-dozen or more. You start at Denver and apply one of the rules. You've moved (perhaps halfway to Wyoming). Now you start at the new point and apply one of the rules. And so on. How do you decide which rule to apply each time? At random. You flip a coin, or you let the computer make a random choice. Each rule has a "weight" that makes it more likely or less likely to be chosen. With six rules of equal weight, you could just roll one die to choose the rule. Or perhaps you would like to give rule No. 1 more weight, make it more likely to be chosen. The computer will handle the random decision-making.

Although the process uses randomness, the outcome is not random at all. The points hop around too rapidly to follow. An image forms on the screen, first faint, then more and more distinct. It is a kind of strange attractor, like the horseshoe attractor of Hénon. By changing the rules, and changing their weights, you can change the image. For any particular set of rules, though, the same image will always appear, like a ship arriving through the mist.

"Randomness is a red herring," Barnsley has said. "The object itself does not depend on the randomness. With probability one, you always draw the same picture. It's giving deep information, probing fractal objects with a random algorithm. Just as, when we go into a new room, our eyes dance around it in some order which we might as well take to be random, and we get a good idea of the room. The room is just what it is. The object exists regardless of what I happen to do."

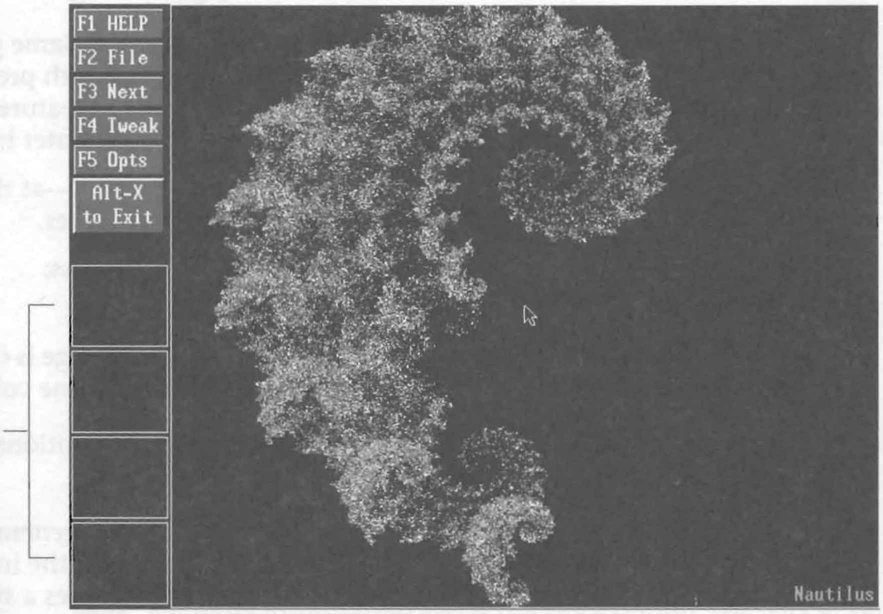
Things To Try

Press N until you see the picture of the Fern. Press W to go into triangle editing mode. Press C several times so you can see images of all the available triangles. Now use the cursor to move the corners of one of the triangles, press C to activate another map for editing, and use the cursor to move the corners of the next triangle. After doing this for a while, press U repeatedly to see the image step back through all the changes you've just done.

Next, press Z repeatedly to randomize the maps being used. Sooner or later you will get an image that looks good. Use triangle editing mode to make it even more attractive. Then save it as a parameter file. If you have a friend with CHAOS: The Software, you can share the image.

Starting the Program

When you start The Chaos Game program, the following screen appears:



Stamp outlines

The image area shows an image named Nautilus, generated with The Chaos Game. The upper-left side of the screen displays The Chaos Game menu buttons. At the lower-left side of the screen, you see four rectangular outlines called *stamps*. For information about stamps, see "Using Stamps" on page 128.

Viewing Preset Images

Keyboard shortcut: N to view next; Shift + N to view previous

The Chaos Game program comes with 21 preset images created using the chaos game. The first image, Nautilus, appears on screen when you start the program.

- Select the F3 Next button or press F3 to view the next preset image.
- Press Shift + N to view the previous image.

Continue through all 21 images. After you view the last one, the program returns you to the first image.

Exploring the Program

Before you learn the details of The Chaos Game program, you can explore a few features by experimenting with preset images. Don't worry if you don't fully understand each feature you use in this section. You'll learn about them in detail later in the chapter.

The changes you make here are temporary—at the end of this section you return to the original preset images.

Follow these steps to try some color options:

1. Display a preset image.
2. Press J to change the way the image is colored. Continue to press J to cycle through the nine coloring types.

Note: For more information about color options, see "Using the Options Menu" on page 137.

In The Chaos Game program, fractals are generated by a flock of points that move around the screen under the influence of two or more rules (also called maps). A map defines a systematic way of moving from an initial position to a new position. The movement of a map can consist of one or more of the following: shrinking, stretching, rotating, flipping, or shearing.

To display graphical representations of each rule or "map" that creates a chaos game image, follow these steps:

1. Press M to display map icons.
2. Notice the shape, angle, and location of each map, and the number of maps in the image.
3. Press N to see the map icons for the next image.
4. Cycle through all the preset images to get a feeling for how maps relate to the image.
5. Press M again to turn off map icons.

Note: You learn more about maps in "Using the Tweak Screen" on page 129. For more information about map icons, see "Map Icons" on page 138.

Each complete turn of the chaos game consists of each flock point randomly selecting a map, moving under the influence of the map, and making a mark at the new position it lands on. As this process is repeated over and over, a fractal image emerges.

To watch the repeated transformation of maps produce a chaos game image, follow these steps:

1. Select the F5 Opts button to display the Options menu.
2. Select the Rectangle button next to the Flock Type option.
3. Select the Off button next to the Trace option.
4. Select the ACCEPT button to accept the changes and exit the Options menu.
5. Press (R) to redraw the image and watch the repeated transformations of the maps produce the image.
6. Press (N) to see the maps produce the next image. Continue to press (N) to view all the preset images.

Follow these steps to reset the Trace and Flock options.

1. Select the F5 Opts button again.
2. Select the Dots button next to the Flock Type option.
3. Select the On button next to the Trace option.
4. Accept the changes to exit the Options menu.

Note: For more information about the Flock option, see "Flock" on page 139. For more information about the Trace option, see "Trace" on page 139.

Try changing the weight distribution in the image. The weight increases or decreases the probability that one map is used instead of other rules.

1. Press (M) to display map icons. Notice the colored circle in the center of each map icon.
2. Move the cursor to a circle at the center of one of the map icon outlines. (If you don't have a mouse, hold down the (Shift) key and press the arrow keys to move the cursor.)
3. Left-click or press (Shift) + (I) to increase the weight of that map. Notice that an asterisk (*) appears in front of the image name to indicate that you've changed the weight.
4. Continue to increase the weight until you see a noticeable difference in the image.
5. Increase the weight of another map in the image.

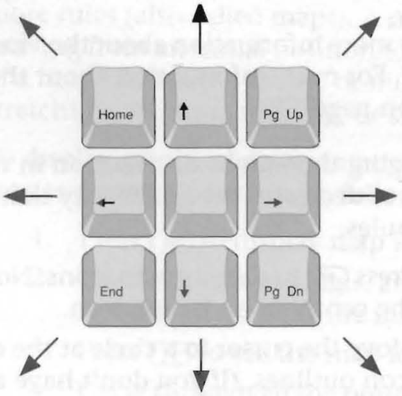
6. Press (V) to undo all weight changes. A message box asks you to confirm that you want to undo all changes.
7. Select the Yes button or press (Y). The image now looks as it did before you made weight changes.
8. Press (M) again to turn off map icons.

Note: For more information about maps and weight, see "Using the Tweak Screen" on page 129 and "Weight Editing Mode" on page 139.

You've completed a quick tour of a few of The Chaos Game features. The following sections provide detailed descriptions of the program's menus and options. Once you understand how to use the program, you can make up your own rules for the chaos game.

Moving Around in the Image Area

You can use the mouse or the keyboard to move the cursor around in the image area. To move around using the keyboard, hold down the (Shift) key and press any key on the numeric keypad. The keys move you in the direction indicated in this illustration:



Numeric keypad keys and direction of cursor movement

Zooming, Panning, and Centering

Zooming In on an Image

CHAOS lets you zoom in on any point in the image area. Zooming is a useful way to explore the self-similarity of chaos game images.

To zoom in, follow these steps:

1. Move the cursor to the location you want. The cursor location becomes the center of the magnified image.
2. Press Ins.
3. Press Ins again to zoom in further.
4. Press Alt + C to quickly restore the image to its original size and location.
5. To zoom out, press Del.

Note: The program runs in a fast *integer mode*. When you zoom in or out far enough, it switches to a slower *real-number mode* and you hear a low-pitched beep. When the program switches back to the faster mode, you hear a high-pitched beep.

To zoom in on a specific area using a zoom box, follow these steps:

1. Press Alt + Z to change the cursor to the zoom box.
2. Hold down the Shift key and move the mouse to resize the zoom box.
   You can also press Shift + ↑ or Shift + ← to make the box larger, and press Shift + ↓ or Shift + → to make the zoom box smaller.
3. Move the zoom box to the location you want to zoom in on.
4. Press Ins or left-click to zoom in on that area.

Panning an Image

When you pan an image, you can see the parts of it that are outside the edges of the image area. Panning is especially helpful when you've zoomed in and can see only a small part of the image on the screen.

1. Move the cursor to the location you want to pan to. (The cursor location becomes the center of the panned image.)
2. Press (Alt)+(O) to pan to the new location.

Centering an Image

As you zoom and pan images, you might feel unsure of where you are in the image. Centering an image moves it to the center of the image area and unzooms it.

- Press (Alt)+(C).

Using Stamps

Keyboard shortcut: (Alt)+(T) to create a stamp;
(Alt)+(1), (Alt)+(2), (Alt)+(3), (Alt)+(4) to select a stamp

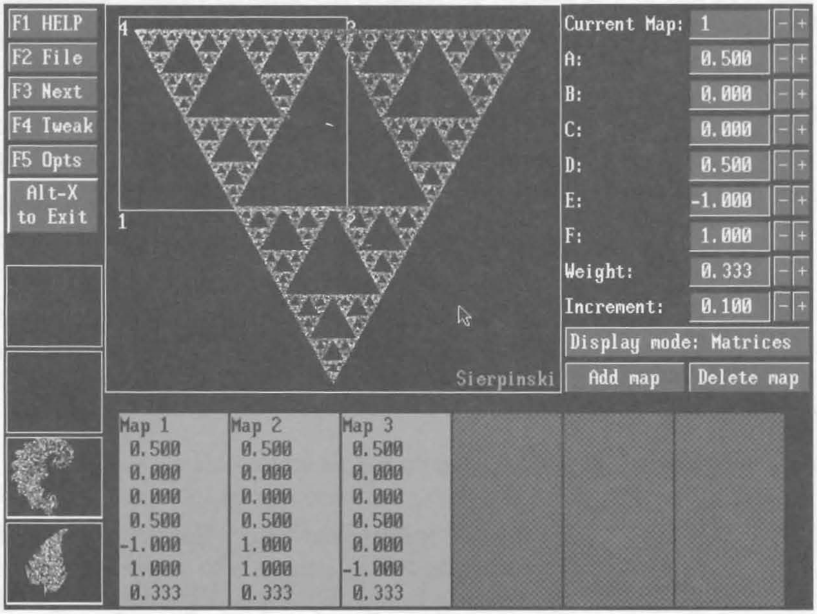
As you experiment with The Chaos Game, you can save a stamp-sized version of up to four images. Then you can select a stamp and quickly redisplay the image in the large image area.

To use stamps, follow these steps:

1. Press (Alt)+(T) to create a stamp of the current image.
2. Display another image and press (Alt)+(T) again to create a second stamp.
3. Select a stamp icon to display that image in the large image area.

Using the Tweak Screen

Tweaking an existing image helps you understand the parameters that make up a Barnsley fractal image. To display the Tweak screen, select the F4 Tweak button.



Current Map: 1
A: 0.500
B: 0.000
C: 0.000
D: 0.500
E: -1.000
F: 1.000
Weight: 0.333
Increment: 0.100

Map 1	Map 2	Map 3
0.500	0.500	0.500
0.000	0.000	0.000
0.000	0.000	0.000
0.500	0.500	0.500
-1.000	1.000	0.000
1.000	1.000	-1.000
0.333	0.333	0.333

Note: The menu buttons are still visible on the left side of the screen. You can use any buttons from the Tweak screen.

Exiting the Tweak Screen

To exit the Tweak screen and return to the main screen, select the F4 Tweak button again.

Understanding Maps

A chaos game image is produced by randomly applying a set of rules over and over again. A rule (also known as a *map*) defines a transformation consisting of one or more of the following: shrinking, stretching, rotating, flipping, or shearing.

In The Chaos Game program, a two-dimensional outline graphically represents each map. To see an example, look at the Sierpinski triangle in the image area of the Tweak screen.

1. Select the F4 Tweak button to display the Tweak screen.
2. Press (N) to flip through the preset images until you reach the Sierpinski triangle image.
3. Notice the outlined rectangle over the upper-left corner of the image. This rectangle represents Map 1 for this fractal.

Each map transforms the entire image area. For example, Map 1 in the Sierpinski triangle reduces the entire image area to 1/2 its size (in both height and width) and moves it up and to the left.

Each map is also defined by six numbers (displayed on the right side of the Tweak screen). These six numbers are used in an algorithm that generates the transformation.

Note: For more information about the mathematical meaning of maps and parameters, see "The Mathematics of the Program" on page 143.

Tweaking Maps

A good way to learn how the parameters affect maps and the resulting fractal image is to experiment with the parameters in an existing image. By adjusting one parameter at a time, you can see the corresponding transformation in the image. You can adjust the parameters in one or several maps and you can add and delete maps to see how that affects the image.

Remember, any changes you make to the 21 images that come with the program are just temporary. And if you're working with a saved version of an image you created, you can always revert back to the saved version if you're not happy with your changes.

Changing the Current Map

Keyboard shortcut: (C)

When you work on the Tweak screen, you can only change one map at a time—the current map. The Current Map button in the upper-right corner of the Tweak screen tells you which map is current. The outline in the image area and the parameters along the right side of the screen represent the current map.

The bottom of the screen displays the parameters that define the other maps in the image. For example, if you're looking at the Sierpinski triangle, you see the parameter numbers for the three maps that generate this image.

Note: If an image has more than six maps, the parameters of the first six maps appear at the bottom of the screen. To see the other maps, left-click the arrow buttons that appear above the first and last map. If you don't have a mouse, move the cursor to the arrow buttons and press ↵.

If you want to tweak one of the other maps in an image, you must first make that map current. To make the next map current

- Select the plus button (+) next to the Current Map button.

The outlined rectangle in the image area and the parameters on the right side of the screen now represent the new current map.

To make the previous map current

- Select the minus button (–) next to the Current Map button.

Note: You can also make a map current by left-clicking in the area at the bottom of the screen that contains the numbers for that map.

Changing the Display Mode

You can tweak map parameters using one of three display modes: Matrices, Triangles, and Angles. The Matrices display mode gives you mathematical control over the changes to the fractal. The Triangles mode lets you adjust the image quickly and get immediate visual feedback. The Angles display mode lets you adjust parameters that have a clear geometric meaning. You'll learn more about these display modes in the next sections.

The current display mode and the corresponding parameter labels appear on the right side of the screen. To change from one display mode to another

- Select the Display mode button.

Suggestion: After you become familiar with the three display modes, you can do an experiment. Use each of the three display modes to try to invert the Sierpinski triangle (that is, make the triangle point up). When the triangle is inverted, the two maps originally at the top of the screen are at the bottom, and the map originally at the bottom of the screen is at the top.

Matrices Display Mode

Matrices mode displays the IFS (Iterated Function System) parameters that define the current map. By changing these numbers (labeled A through F), you can change the current map and the entire image. The A and D parameters tend to change the height and width of the map; the B and C parameters tend to squash or stretch.the map; and the E and F parameters change the location of the map.

IFS parameters are commonly used to publish a description of a chaos game fractal and are a convenient way to exchange fractal images with your friends. If you know the IFS parameters for any fractal, you can use The Chaos Game program to quickly reproduce that fractal on screen.

Note: For information on how to change parameters, see "Changing Parameters" on page 133. For information about the mathematical meaning of the parameters, see "The Mathematics of the Program" on page 143.

Triangles Display Mode

In Triangles display mode, you change maps by changing the X and Y coordinates of each corner in a triangle. You can change the angles of the triangle by changing the coordinate numbers (as described in "Changing Parameters" on page 133) or you can use the mouse or keyboard to move the corners of a triangle (as described in "Triangle Editing Mode" on page 140).

Of the three display modes, Triangles is the only one that lets you change maps from both the Tweak screen and the main image screen. For information on working with triangles from the main screen, see "Triangle Editing Mode" on page 140.

Angles Display Mode

In Angles display mode, each map is defined by six parameters that represent geometric changes to the map on both the x axis and y axis. These six parameters are described as follows:

X-rotation — The angle of the top and bottom sides of the map in relation to the horizontal axis. This parameter is specified in degrees (from –180 to 180).

Y-rotation The angle of the right and left sides of the map in relation to the vertical axis. This parameter is specified in degrees (from –180 to 180).

X-scale The change in size of the map in the horizontal direction. The acceptable range for this parameter is -100 to 100.

Y-scale The change in size of the map in the vertical direction. The acceptable range for this parameter is -100 to 100.

X-trans The change in location of the map in the horizontal direction. This parameter is identical to the E parameter in Matrices display mode. The acceptable range for this parameter is -100 to 100.

Y-trans The change in location of the map in the vertical direction. This parameter is identical to the F parameter in Matrices display mode. The acceptable range for this parameter is -100 to 100.

Changing Parameters

You can change parameters in increments or you can directly enter a new parameter value.

Changing Parameters in Increments

You change parameters in the increment displayed in the Increment button. To use a different increment, see "Changing the Increment" on page 134.

Note: In Angles display mode, the X-rotation and Y-rotation parameters change in an amount equal to ten times the increment. For example, if the increment is 0.1, the rotation parameters change by 1 degree.

- Select the plus button (+) next to any parameter to increase the parameter value.
- Select the minus button (–) next to any parameter to decrease the parameter value.

Entering a New Value for a Parameter

To type in a new value for a parameter

- Enter the new value in the parameter number entry field and press ↵.

Suggestion: To understand how changing parameters affects the fractals, experiment with changing different parameters in the preset images.

Changing the Increment

The Increment button controls how much of a change the plus (+) and minus (–) buttons make for the parameters (including weight). You can set the increment to 1.000, 0.100, 0.010, or 0.001, depending on the precision you want to use.

- Select the plus button (+) next to the increment to increase the increment.
- Select the minus button (–) next to the increment to decrease the increment.
- Left-click on the Increment number field to quickly cycle through the available increments using a mouse.

Randomizing Map Parameters

Keyboard shortcut: Z

Another quick and often interesting way you can change map parameters is by randomization. By assigning random parameters to the existing maps in an image, you completely transform the image. The results you get when you randomize are impossible to predict. Sometimes the image appears as randomly distributed dots, and sometimes interesting shapes appear.

You can randomize an image from the Tweak screen or from the large image screen.

To randomize the maps in an image, follow these steps:

1. Press Z.
2. Continue to randomize the maps by pressing Z again and again.

Suggestion: If you discover a random image that you like, be sure to save the parameters. That way, you can load the image again or share it with friends. For information on saving parameter files, see "Saving Parameters" on page 211.

After you've randomized a preset image, you can easily restore the original image by following these steps:

1. Press N to display the next image.
2. Press Shift + N to return to the original preset image.

Changing the Weight or Probability

Each map in a chaos game image is assigned a weight. The weight represents the probability that a map will be used (relative to the use of the other maps). When you add weight to one map, you reduce the weight of the other maps: the total weight of all maps must add up to 1. Changing the weight of the maps can have a dramatic effect on the resulting image.

To increase the weight of a map

- Select the plus button (+) next to the Weight parameter.

To decrease the weight of a map

- Select the minus button (–) next to the Weight parameter.

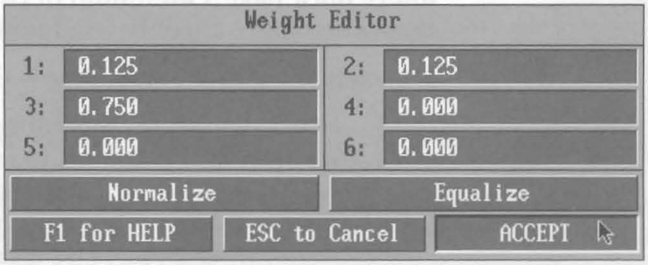
Note: For information about changing weight from the main screen, see "Weight Editing Mode" on page 139. For information about the most precise way to change the weight for one or more maps, see the next section.

Using the Weight Editor

Keyboard shortcut: E

The Weight Editor gives you the most precise way to change the weight of one or more maps in an image. You can display the weight editor from any screen or editing mode in The Chaos Game.

- Press E. The Weight Editor dialogue box appears:



Weight Editor

1: 0.125 2: 0.125

3: 0.750 4: 0.000

5: 0.000 6: 0.000

The Weight Editor shows the current weight of each map in the image. You can change the weight of any map by entering a new weight value for that map.

- Enter the new weight in one of the weight entry fields and press ↵.

The weights of all maps in a chaos game image must equal 1. If you change the weight of one map, the program will adjust the weights of all maps so they add up to 1. You can use the Normalize button to see the adjusted weight values before you exit the Weight Editor.

The Equalize button lets you quickly assign equal weight to all maps in the image.

Adding Maps

Keyboard shortcut: A

You can change the fractal by adding a new map. The Chaos Game allows up to 20 maps in an image.

- Select the Add Map button to add a map.

The new map is a small rectangle in the center of the image. The new map is the current map and you can tweak it as you would any other map.

When you add a new map, the weights of all maps in the image are equalized. For more information about equalizing weights, see "Using the Weight Editor" on page 135.

Note: If you add several maps in quick succession, the new maps will initially be on top of each other.

Deleting Maps

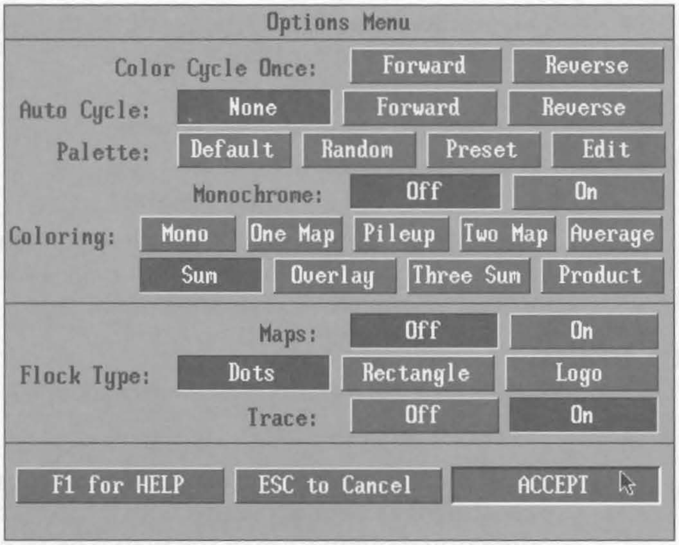
You can change the fractal by deleting an existing map. Each image must have a minimum of two maps, so you can't delete a map when there are only two maps.

To delete the current map, follow these steps:

1. Select the Delete Map button. A message box asks you to confirm that you want to delete the map.
2. Select the Yes button or press Y to confirm the delete.

Using the Options Menu

You can use the Options menu from the main screen in The Chaos Game or from the Tweak screen. To display the Options menu, select the F5 Opts button or press F5.



Color Options

The following color options are common to five of the six CHAOS programs and are described in "Using Color Options" on page 214.

- Color Cycle Once
- Auto Cycle
- Palette (Default, Random, Preset, Edit)
- Monochrome

Coloring

Keyboard shortcut: J cycle forward; Shift + J cycle backward

The Chaos Game program lets you choose from various ways of coloring the fractals. Different coloring types look best with different images. You can cycle through the coloring types to find the one you like best for an image.

To change from one coloring option to the next, select one of the following coloring type buttons on the Options menu.

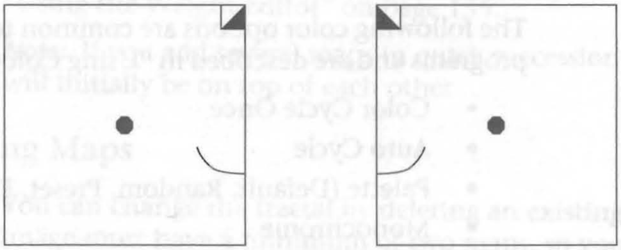
- Mono
- One Map
- Pileup
- Two Map
- Average
- Sum
- Overlay
- Three Sum
- Product

Note: For technical details on these coloring options, see "The Mathematics of the Program" on page 143.

Map Icons

Keyboard shortcut: M

In The Chaos Game, you can turn map icons on and off. Map icons are shown as parallelograms with a "face" that shows you the directional orientation of the map. The face has a "mouth" and an "eye" located in one corner of the map. The following illustration shows a map facing right and a map facing left.



facing right   *facing left*

The outline of a map indicates its relative weight. The more solid the line, the greater the weight of that map. Each map also has a small octagon at its center. If a map has zero weight, it appears as a hollow octagon: the border of the parallelogram is invisible.

- Select the On or Off button next to Maps on the Options menu to toggle map icons on and off.

Flock

Keyboard shortcut: F

Images in The Chaos Game program are generated by a flock of points that appear in locations determined by the maps. The program gives you the following three flock options:

- Dots—paints the image with flocks made up of sixteen equally spaced points. The default flock type is Dots.
- Rectangle—paints the image with flocks made up of the rectangular outlines of the maps.
- Logo—paints the image with flocks that look like the Autodesk logo.

To switch between the three available flock options

- Select the Dots, Rectangle, or Logo button next to the Flock option on the Options menu.

To redraw the image using the current flock

- Press R.

Note: For more information on flocks, see "The Mathematics of the Program" on page 143.

Trace

Keyboard shortcut: T

If trace is on (the default setting), each flock point leaves a permanent mark. If trace is off, then only the current position of each flock point is shown.

- Select the On or Off button next to Trace on the Options menu to toggle trace on and off.

Weight Editing Mode

Keyboard shortcut: Shift + I to increase; I to decrease

When you start The Chaos Game program, you're in *weight editing mode* and the cursor is an arrow. Weight editing mode lets you easily change the weight of the maps in the fractal. As you add or remove weight from one map, the weight of the other maps is adjusted.

- To increase the weight of the map whose center is nearest the cursor, left-click.
- To decrease the weight of the map whose center is nearest the cursor, right-click.
- To remove all weight from the map whose center is nearest the cursor, press (K).

You can use weight editing mode at the main screen (with the large image area) or from the Tweak screen.

When you're in weight editing mode, you can also use any other keyboard shortcut to change the fractal. For example, you can change the coloring, display the map icons, or add a new map.

Note: For other ways of changing weight in The Chaos Game, see "Changing the Weight or Probability" on page 135 and "Using the Weight Editor" on page 135.

Triangle Editing Mode

In *triangle editing mode*, each map is represented by a triangle that is one half of the parallelogram that describes the map. You can quickly change a fractal by changing the shape of a triangle. You can use triangle editing mode from the main screen (with the large image area) or from the Tweak screen.

At the main screen, you have the option of adjusting the corners of the triangles using the mouse or keyboard. From the Tweak screen, you have the additional option of entering new X and Y coordinate values. For more information about entering new coordinate values at the Tweak screen, see "Changing Parameters" on page 133.

Starting Triangle Editing Mode

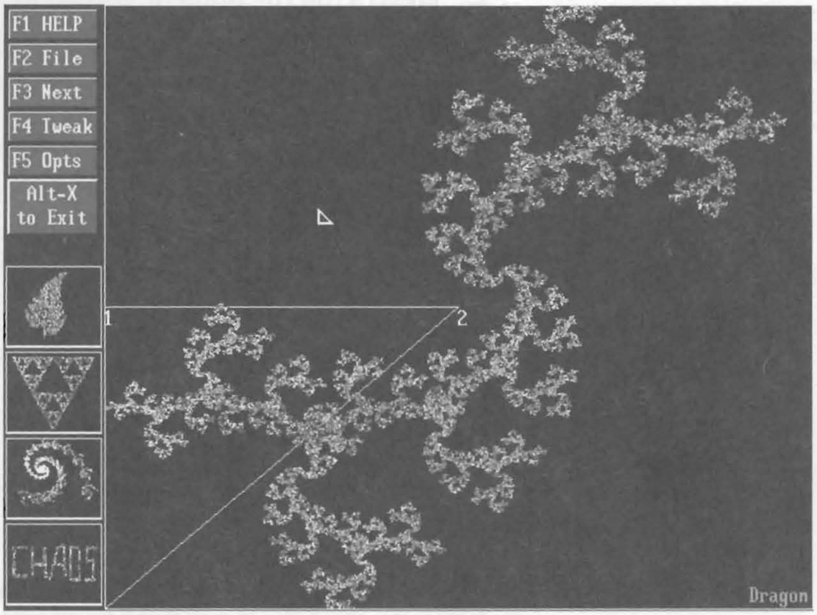
To toggle between weight and triangle editing mode from the main screen

- Press (W).

To use triangle editing mode from the Tweak screen

- Select the Display Mode button until Triangles appears.

When you are in triangle editing mode, the current map is represented by a triangle with numbered corners and the cursor is a small triangle.



Editing Triangles

You can change the shape of the current map by moving one or more corners of the triangle with the mouse or keyboard.

To change the current triangle using a mouse, follow these steps:

1. Left-click to move the closest corner to the location of the cursor.
2. Continue to move the corner with a series of small moves. (Because the closest corner moves to the cursor location, a series of small moves helps prevent you from moving the wrong corner.)

Note: You can also hold down the left mouse button and drag the corner to a new location.

To change the current triangle using the keyboard, follow these steps:

1. Hold down the Shift key and use the arrow keys to move the cursor to the new location for the corner. (If the new location is close to another corner, you'll have to make the move in small increments.)
2. Press 5 on the numeric keypad to move the corner to the cursor location.

If you're using triangle editing mode from the Tweak screen, the program updates the corner parameters on the right side of the screen and the IFS parameters displayed at the bottom of the screen.

Suggestion: You can often get interesting results by pulling a corner through the opposite side of the triangle, effectively "flipping" the triangle.

As you edit triangles, it's helpful to turn on the display of map icons:

- Press M.

You can also quickly make the next map current:

- Right-click or press C.

And you can easily undo changes you make in triangles editing mode:

- Press U. (You can continue to press U to undo your most recent changes one at a time, up to a maximum of 100 changes.)

To undo all changes, follow these steps:

1. Press V. A message box appears asking if you want to undo all changes.
2. Select the Yes button or press Y to confirm that you want to undo all changes.

Saving and Loading Files

You can use the options on the File menu to save and load parameter files and *.gif* files. For more information, see "Saving and Loading Files" on page 209.

The Mathematics of the Program

In The Chaos Game, we see fractals that are generated by having a flock of points hop around on the screen under the influence of two or more maps. Why such a simple technique can produce such intricate fractals is initially a bit of a mystery—but as you play with the program, you should eventually begin to understand.

The weighted set of maps being used is known as the iterated function system, or IFS for short. Each complete *turn* of the chaos game consists of having each flock point randomly select one of the current IFS maps, move under the influence of this map, and make a mark at the new position it lands on. If trace is on (as it usually is), each flock point leaves a permanent mark each time it touches down. If trace has been toggled off with the T key, then only the current position of each flock point is shown.

There are three kinds of flocks you can use: Dots, Rectangle, or Logo. The Dots flock starts out as sixteen points, equally spaced across the image area screen in a four by four grid. The rectangle flock starts out as a frame rectangle of about 2,000 dots placed one per pixel around the perimeter of the image area. The logo flock starts as a block of about 2,500 dots grouped together to make an Autodesk logo in the upper-right region of the image area. It is interesting to experiment on a simple IFS, such as the Fern, trying different combinations of trace and flock. Pressing R clears the screen and refills a flock from scratch.

There are an endless number of iterated function systems, or weighted sets of maps, which you can use. An individual map defines a systematic way of moving from any initial plane position (x,y) to a new plane position (newx,newy). The maps we use in this program are all *bilinear* or *affine* maps, meaning that each of them can be defined using two linear equations in x and y. The equations defining a single map have the form:

```
newx = a * x + b * y + e;

newy = c * x + d * y + f;
```

The effect of such a map is always to move the corners of any rectangle in the plane into the corners of a parallelogram somewhere else in the plane. Moreover, these maps act in an orderly fashion on the points on a rectangle's edge. That is, if an edge point is halfway between two corners, its image under the map will be halfway between the images of the corners. Such affine maps, defined in terms of two linear equations, cannot warp a rectangle into an irregular trapezoid, nor can they bend a straight side into a curved arc. The complete inventory of what an affine map can do to a rectangle is to shrink or stretch it either vertically or horizontally, to rotate its horizontal or vertical edges, or bodily to move the whole rectangle from here to there. If an affine map happens to rotate one of the edges right through one of the other edges, it will also have the effect of performing a mirror flip on the rectangle.

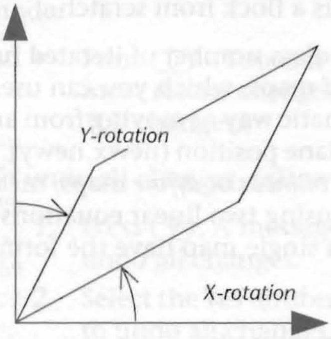
Sometimes an affine map's two linear equations are written as a single matrix equation:

$$\begin{bmatrix} newx \\ newy \end{bmatrix} = \begin{bmatrix} \mathrm{a} & \mathrm{b} \\ \mathrm{c} & \mathrm{d} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \mathrm{e} \\ \mathrm{f} \end{bmatrix}$$

If you look at the Tweak menu, you will see the parameters a, b, c, d, e, and f for each map listed at the bottom of the screen. If the Tweak menu Display Mode is set to Matrices, then you can directly change the values of the matrix parameters a, b, c, d, e, and f. Changing e and f simply moves the image rectangle about, but the effects of a, b, c, and d are harder to disentangle.

Another way of characterizing an affine map is to find parameters X-scale, Y-scale, X-rotation, and Y-rotation such that

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} (x\text{-}scale)\cos(x\text{-}rotation) & -(y\text{-}scale)\sin(y\text{-}rotation) \\ (x\text{-}scale)\sin(x\text{-}rotation) & (y\text{-}scale)\cos(y\text{-}rotation) \end{bmatrix}$$



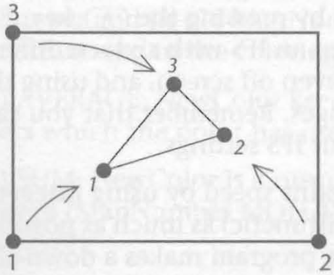
X-scale is the factor by which the rectangle is stretched or shrunk along the horizontal axis, and Y-scale is the factor by which the rectangle is stretched or shrunk along the vertical axis. A scale factor of 0.5 means, for instance, that a distance is made half as big; a scale factor of 3.0 would mean that a distance is to be made three times as large. X-rotation is the number of degrees through which the rectangle's horizontal lines are rotated, and Y-rotation is the number of degrees through which the rectangle's vertical lines are rotated. Positive rotation is counterclockwise and negative rotation is clockwise, and X and Y rotation are measured, respectively, from the horizontal and the vertical positions. In this context, the amounts e and f by which the whole rectangle is moved (or *translated*) can be renamed X-trans and Y-trans.

If one knows where an affine map moves one specific rectangle, then one can actually compute the values of a, b, c, d, e, and f, and then one will know where the map moves all rectangles, and all points, of the plane. This is the basis of the pictures of the maps that appear as white rectangles on the Tweak menu, and is also the basis of the faint colored pictures of the maps which you can toggle on by pressing the M key. In each case, the map pictures are images of the *frame rectangle* that makes up the perimeter of the starting image.

Just by looking at a parallelogram, it is sometimes hard to tell what combination of rotations and mirrorflips the depicted map is supposed to perform. So that the eye can easily pick this out, our colored map images act as if the original frame rectangle has an *eye* at its northeast corner, and a *mouth* on its east side. This image is modelled on the famous A Square, hero of Edwin Abbott's classic book *Flatland*.

Yet another way of specifying a map can be arrived at by noting that if we know the image position of three of a parallelogram's corners, then we can compute the location of its fourth corner. So if we specify where three of some specific rectangle's corners are to be moved by an affine map, then we have also effectively specified where the fourth corner must go, and since we then know where one rectangle is taken, we know where all rectangles and all points are taken.

In the Tweak menu, if the Display Mode is set to Triangles, then you can describe a map by specifying the X and Y coordinates of the images of the starting rectangle's three corners: Corner 1, Corner 2, and Corner 3.



In the Tweak menu or on the main screen, you can toggle on a graphic map-editing mode by pressing the (W) key. In this case the map being edited is shown as a white triangle. You can alter the map simply by dragging the corners of the triangle around. Pleasing effects can often be achieved by dragging one of a triangle's corners through one of the sides, thus incorporating a mirror flip into the affine map being edited.

In selecting which map to obey, a flock point uses a weighted randomizer rather than a pure randomizer. That is, if there are N maps, the different maps need not necessarily all have the same likelihood of 1/N of being picked. The weight of each map is expressed as a probability between 0 and 1. The map weights can be adjusted numerically on the Tweak menu, or they can be graphically changed using the cursor. As all the separate weights must jointly sum up to 1, changing one map's weight must inevitably affect the weights of some of the other maps.

The basic principle in weighting the maps is that, if one wants an evenly lit fractal image, then each map's weight should be proportional to the area of its colored map image. That is, a map which only shrinks the frame rectangle a little should have a high weight, and a map which shrinks the frame rectangle a lot should have a low weight. The colored map images make the weights visible by having the brightness of the map's edges be proportional to the map's weight. If a map has no weight at all, we indicate its location by showing a small hollowed-out octagon.

Sometimes, as in our Cloud or Mud IFS images, one does not want the fractal to be evenly lit. In the case of the Cloud, for instance, the weights are tweaked so as to make the points land on the bottom half of the cloud more often than on the top, with the goal of making it look as if the sun is setting somewhere below the edge of the screen.

A different use of weight changes is to bring various maps in and out of play in a continuous way. In our Nautilus IFS, for instance, there are six maps in the IFS, and moving the weights around causes dramatic changes in the image. You can always add more maps to an IFS by pressing the (A) key. Eventually you might enjoy setting up an IFS with sixteen differently tweaked maps, some of them even off screen, and using this to explore the effects of weight changes. Remember that you can use the File menu to save any of your IFS settings.

Our program gains speed by using integer arithmetic (instead of real number arithmetic) as much as possible. If you zoom in or out far enough, the program makes a down-sliding sound to signal that it has switched to the slower real-number mode. When you zoom back into the range where integer accuracy suffices, the program makes an up-sliding noise. If you zoom in very far, you will notice that it takes a long time for the image to become distinct. This is because the Chaos Game computation is essentially *holistic* in the sense that no piece of these fractals can be computed without computing the whole thing.

Now we'll explain the methods by which colors are assigned to the positions where the flock points land. When a point moves, it uses a randomizer to pick a MapNumber, giving the number of the Barnsley map to be moved by. This map gives the point a new screen position to land on. Before it lands, the point can look at the color presently on the screen, which we call ScreenColor. In addition, each flock point has a small memory area in which it can keep a record of its last few MapNumbers. So the point has three pieces of information: its MapNumber, the ScreenColor, and its memory of its past MapNumbers. Based on these three numbers it computes a NewColor.

Keep in mind that our programs use only sixteen colors, so anytime a computed NewColor is greater than 16, we replace NewColor by the NewColor MOD 16, which is the remainder you get when you divide NewColor by 16. The coloring styles work as follows.

1) MONO: NewColor is always the maximum palette color, which is usually white.

2) ONE MAP: NewColor is the same as the MapNumber, the number of the map which just moved the point. This coloring style makes it clear what the different maps do.

3) PILEUP: NewColor is the present ScreenColor of the target pixel plus one. The more points that land on a pixel, the higher its color number becomes—until the numbers get larger than 16 and wrap around. This coloring style gives you an idea of the frequency with which points are hitting certain parts of the image.

4) TWO MAP: NewColor is based on the sum of the point's MapNumber and its old MapNumber. The effects of the last two maps are thus seen in the coloring.

5) AVERAGE: NewColor keeps a running average of the MapNumbers which the point has used.

6) SUM: NewColor is a running sum of the point's successive values of (MapNumber MOD 4).

7) OVERLAY: NewColor is gotten by XORing together the MapNumber and the last two values of the point's MapNumbers. To "XOR" two numbers means to compare their binary bits and put zeroes in the places where the bits are the same, and ones where the bits are different. XORing numbers is a good way of preserving as much of their information as possible.

8) THREE SUM: NewColor is the sum of the last three MapNumbers.

9) PRODUCT: NewColor is the product of the last three MapNumbers.

Many other coloring styles can be imagined. The ones we used here were arrived at partly by trial and error. Some of the effects they produce are rather surprising. Certain styles look better with certain setups. When you have a new image, it's a good idea to press the (J) key many times to see how the image looks with the various color styles. Note that the effects of some of the coloring styles change if you press (Alt)+(E) to extend the color palette.

Chapter 6
Fractal Forgeries

Clouds are not spheres, Mandelbrot is fond of saying. Mountains are not cones. Lightning does not travel in a straight line. The new geometry mirrors a universe that is rough, not rounded, scabrous, not smooth. It is a geometry of the pitted, pocked, and broken up, the twisted, tangled, and intertwined. The understanding of nature's complexity awaited a suspicion that the complexity was not just random, not just accident . . . Mandelbrot's work made a claim about the world, and the claim was that such odd shapes carry meaning. The pits and tangles are more than blemishes distorting the classic shapes of Euclidean geometry. They are often the keys to the essence of a thing.

What is the essence of a coastline, for example? Mandelbrot asked this question in a paper that became a turning point for his thinking: "How Long is the Coast of Britain?"

Mandelbrot had come across the coastline question in an obscure posthumous article by an English scientist, Lewis F. Richardson, who groped with a surprising number of the issues that later became part of chaos . . . Wondering about coastlines and wiggly national borders, Richardson checked encyclopedias in Spain and Portugal, Belgium and the Netherlands and discovered discrepancies of twenty percent in the estimated lengths of their common frontiers.

Mandelbrot's analysis of this question struck listeners as either painfully obvious or absurdly false . . .

In fact, he argued, any coastline is—in a sense—infinitely long. In another sense the answer depends on the length of your ruler. Consider one plausible method of measuring. A surveyor takes a set of dividers, opens them to a length of one yard, and walks them along the coastline. The resulting number of yards is just an approximation of the true length, because the dividers skip over twists and turns smaller than one yard, but the surveyor writes the number down anyway. Then he sets the dividers to a smaller length—say, one foot—and repeats the process. He arrives at a somewhat greater length, because the dividers will

> *capture more of the detail and it will take more than three one-foot steps to cover the distance previously covered by a one-yard step. He writes this new number down, sets the dividers at four inches, and starts again. This mental experiment, using imaginary dividers, is a way of quantifying the effect of observing an object from different distances, at different scales . . .*
>
> *. . . Mandelbrot found that as the scale of measurement becomes smaller, the measured length of a coastline rises without limit, bays and peninsulas revealing ever-smaller subbays and subpeninsulas—at least down to atomic scales, where the process does finally come to an end. Perhaps. (*Chaos: Making a New Science, *pages 94–96.)*

An infinitely long line crowded into a finite space is more than a line, yet less than a plane. Mandelbrot expresses this by saying that such a line has a fractional dimension which lies between one and two. Similarly a mountain range covered with peaks covered with jagged rocks is more than a plane, and is accordingly said to have a fractional or *fractal* dimension between two and three.

By mimicking the shrinking-dividers experiment on maps and photographs, scientists have actually arrived at rough estimates of the fractal dimensions of such things as seacoasts, river networks, tree bark, and galaxies. These measurements do not provide an explanation for these structures; they are simply a new way of classifying them. The explanations lie hidden in the chaotic dynamics of the motions that create natural forms.

Computer graphics artists are perfectly willing, however, to short circuit the process of explanation. If, for instance, the Rocky Mountains resemble a fractal of dimension 2.61, then why not just build such a fractal next time you need a mountain range?

The process of building a fractal of a desired dimension involves synthesizing an image by adding increasing amounts of detail—the fractal aspect is that at each level one adds more detail than might strictly be expected. Line segments get kinks in them, triangular faces acquire humps, and so on. The first programs did this in an iterative, face-by-face way; our Fractal Forgeries program uses a more sophisticated technique that adds new levels of detail to all parts of the image at once.

The computer artist Loren Carpenter generated the fractal alien landscapes that Lucasfilm supplied for the movie *Star Trek II*. Describing his experience in *The College Mathematics Journal*, Carpenter writes:

> *"One of the major problems with fractals in synthetic imagery is the control problem. They tend to get out of hand. They will go random all over the place on you. If you want to keep a good tight fist on it and make it look like what you want it to look like, it requires quite a bit of tinkering and experience to get it right. There are not many people around who know how to do it."*

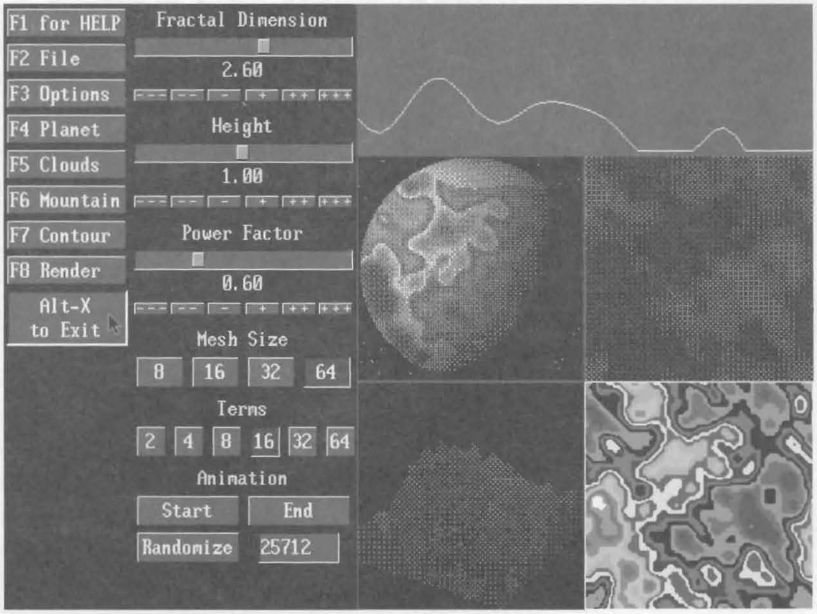
Using our Fractal Forgeries program, you can have the fun of learning how to interactively adjust the fractal dimension to produce landscapes, clouds, or planets that look the way you want.

Things To Try

Take a single setting of the parameters and draw it as Mountain, Cloud, Planet, and Contour in the four image boxes. Take one type of fractal, say Mountain, twiddle the parameters till it looks decent, and then see how it looks over the different range of fractal dimension settings. In the process of adjusting parameters, don't neglect to use the Randomize button to see different realizations of the same parameter settings. Once you find a parameter setting which gives interesting changes, make a ten-frame animation of it, so you can see dynamically the effects of changing the fractal dimension.

Starting the Program

When you start the Fractal Forgeries program, you see the following screen:


Forgery parameters *One-dimensional representation*

Fractal Dimension 2.60
Height 1.00
Power Factor 0.60
Randomize 25712

The upper-left side of the screen displays the menu buttons. To the right of the menu buttons are forgery parameters (three slider bar options and four sets of buttons). In the upper-right corner of the screen, you see a contour line that is a one-dimensional representation of the current parameters. At the lower-right side of the screen, you see four sample forgery images generated from a single set of parameters.

Exploring the Program

Before you learn all the details of Fractal Forgeries, you can quickly explore a few features and generate completely new pictures. Don't worry if you don't fully understand each feature you use. You'll learn about each parameter and option in detail later in this chapter.

The opening screen displays several parameters that you can adjust to create unique forgeries. You'll learn about these parameters in the next section "Changing Forgery Parameters" on page 155. For now, you'll use the Randomize option to create a completely new image.

To randomly create a new mountain landscape, follow these steps:

1. Select the Randomize button. The program changes the number to the right of this button. The contour line at the top of the screen also reflects the change.
2. Select the F6 Mountain button. The program generates the image and displays it in the currently selected (upper-left) image area.

Note: You can repeat steps 1 and 2 as often as you like.

The Fractal Forgeries program generates up to four different types of images from a single set of parameters. Without changing anything, you can create planets, mountains, cloud formations, and contours.

To create a planet without changing parameters, follow these steps:

1. Select the upper-right image area by left-clicking in that image box or pressing (2).
2. Select the F4 Planet button. The program generates the image and displays it in the upper-right image area.

Note: You can also generate cloud and contour images from these parameters by selecting the F5 Clouds and F7 Contour buttons. For more information, see "Displaying Forgeries" on page 159.

In addition to the parameters on the opening screen, you can use options to change specific image types.

To use options to change the planet image, follow these steps:

1. Select the F3 Options button. The Options menu appears.
2. Select the Planet Options button. The Planet Options menu appears.
3. Enter **3** in the Hour field to create a planet with a rotation shadow set to three o'clock.
4. Select the ACCEPT button to confirm your change and exit the menu.
5. Select the lower-left image area by left-clicking in that image box or pressing (3).

6. Select the F4 Planet button. The program generates the planet image with a rotation shadow and displays it in the lower-left image area.

Note: For information on all of the options available in the program, see “Using the Options Menu” on page 160.

Once you generate an interesting image, you can display a full-screen rendering of the image. If you have a VGA monitor, you can display the rendering in 256 colors. (The images on the main screen are limited to 16 colors.)

To render an image, follow these steps:

1. Select the image you want to render by left-clicking in the image area or pressing ①, ②, ③, or ④ to move to the correct image area.
2. Select the F8 Render button. The Render an Image menu appears.
3. Select the button labeled with the type of image you want to render. For example, if you’re rendering a planet, select the Planet button.
4. If you have a VGA monitor, select the 256 colors button to render in the highest available resolution.
5. Select the Render to Screen button.
6. Select the RENDER button. The program displays a full-screen rendering of the image you selected.
7. Click either mouse button or press any key to return to the main program screen.

Note: For more information on rendering an image, see “Rendering an Image” on page 166.

You can also create and play back animations of the forgeries you generate in Fractal Forgeries. Before you create an animation, you should become familiar with the available parameters and options. For more information, see “Animating an Image” on page 168.

Changing Forgery Parameters

The Fractal Forgeries program generates pictures of planets, clouds, mountains, and contours from one set of calculated data or parameters. But each type of image looks best with a different combination of parameter settings. And although each parameter has a different effect on the images, the parameters interact in subtle ways. By experimenting with many different settings, you'll develop an intuitive understanding of the relationship between these parameters and the images they create.

Note: For information on the mathematical meaning of the forgery parameters, see "The Mathematics of the Program" on page 171.

One-Dimensional Representation of Parameters

The contour line in the upper-right corner of the screen is a one-dimensional representation of the current parameter settings. As you change the forgery parameters, the line changes to show the relative irregularity or smoothness of the resulting image. This line is not an outline of the resulting image, but it gives you a general impression of the quality of the resulting landscape.

Changing the Fractal Dimension

The fractal dimension determines the degree of roughness of the terrain or corresponding aspects of other images. A higher fractal dimension produces a more irregular or bumpy surface.

Fractal dimensions are not limited to integers as they are in Euclidean geometry. In Fractal Forgeries, you can set a fractal dimension anywhere between 2.0 and 3.0. For example, you can use a fractal dimension of 2.5 to create an image that uses more than 2-dimensional space but less than 3-dimensional space.

To change the fractal dimension using a mouse

- Hold down the left mouse button and drag the box in the slider to the position you want.

To change the fractal dimension using the keyboard, follow these steps:

1. Press F to move the cursor to the Fractal Dimension slider bar.
2. Press ← or → to move the slider. Press Shift + ← or Shift + → to move the slider in larger increments.

Note: You can also select the plus or minus buttons below the slider bar to change the fractal dimension. The greater the number of plus or minus symbols on the button, the larger the increment.

Changing the Height

You can change the height of a forged planet or mountain landscape from 0 (water level) to 2.00 (very high elevations). Low numbers generate swamp and lake terrain; high numbers generate mountain ranges. With the height set to zero, the image is all water. At a height of 2.00, the landscape is mountains with very tall peaks. Changes in Height affect only Planets and Mountains—not Clouds and Contours.

Note: To change the height using the keyboard, press H to move the cursor to the Height slider bar. Use the slider bar as described in the previous section "Changing the Fractal Dimension" on page 155.

Changing the Power Factor

The power factor scales height variations. The power factor has the most dramatic effect on mountain images. In mountain forgeries, power factors greater than 1.0 re-create the effects of erosion by flattening the lower elevations while leaving steep spires. Power factors less than 1.0 have the opposite effect—they flatten the peaks and increase steepness at water level. A power factor of zero creates a plateau effect. You can set a power factor from zero to 2.0.

Clouds tend to look better when the Power Factor is set to a low number.

Note: To change the power factor using the keyboard, press P to move the cursor to the Power Factor slider bar. Use the slider bar as described in "Changing the Fractal Dimension" on page 155.

Changing the Mesh Size

Keyboard shortcut: (Shift)+(M) to increase; (M) to decrease

The program generates images by preparing a matrix of random data. You can change the size of this matrix by changing the mesh size. Increasing the mesh size means you calculate the image at more points, which produces a more realistic picture. Using a high mesh size increases the calculation time.

To change the mesh size

- Select the button for the mesh size you want to use.

Suggestion: If you're randomly creating images, you might want to use a low mesh number to save time. Then, when you find an image you like, increase the mesh size.

Changing the Terms

Keyboard shortcut: (Shift)+(T) to increase; (T) to decrease

The Terms parameter controls how many Fourier sum terms are used in the algorithm that generates the forgery. (For information about the Fourier sum and the mathematics used in the Fractal Forgeries program, see "The Mathematics of the Program" on page 171.) Reducing the number of terms makes the surface smoother.

To change the terms

- Select the button for the number of terms you want to use.

Setting Animation Parameters

Keyboard shortcut: (S) to set start parameters;
(E) to set end parameters

If you have a VGA monitor, you can create an animation of a forgery. The animation shows a forgery transforming from an image based on one set of parameters to an image based on another set of parameters. For example, you can create an animation of a mountain forgery that starts with a height of zero (all water) and ends with a height of 2.00 (tall peaks). When you play back the animation, you'll see tall peaks rise out of the water.

You can set starting and ending animation parameters from the main screen or from the Animation dialogue box.

Suggestion: You might want to read "Animating an Image" on page 168 before setting starting and ending parameters for the first time.

To set starting animation parameters from the main screen, follow these steps:

1. Set the parameters for the first image in the animation.
2. Select the Animation Start button at the bottom of the screen or press S. The selected parameters are now saved as the starting parameters for the animation.

To set ending animation parameters from the main screen, follow these steps:

1. Set the parameters for the final image in the animation.
2. Select the Animation End button at the bottom of the screen or press E. The selected parameters are now saved as the ending parameters for the animation.

Note: When you're ready to create the animation, see "Animating an Image" on page 168.

Randomizing the Forgery

Keyboard shortcut: Z

The Randomize option lets you change the *seed* or start value used by the pseudo-random number generator that creates data for the image. Each seed value creates a completely different forgery. You can re-create a forgery by using the same seed and parameters.

You can select the Randomize button to let the program choose a random seed, or you can enter a specific seed value.

To use a random seed value

- Select the Randomize button.

To enter a specific seed value

- Enter a number from **0** to **32767** in the entry field next to the Randomize button and press ↩.

Displaying Forgeries

After you change the forgery parameters, you can quickly display low-resolution versions of up to four images. To display a forgery using the current parameter settings, you select an image area, and then select the menu button for the type of forgery you want to generate.

Selecting the Image Area

Keyboard shortcut: 1, 2, 3, or 4

The currently selected image area is shown by the highlighted border around the area. When you create a new forgery, the image is displayed in the selected image area. If an image is already in the selected box, the new image replaces it.

To select the image area where the next forgery will appear

- Left-click in the image box or press 1, 2, 3, or 4 to select one of the four image areas. The highlighted border moves to the image area you selected.

Selecting the Forgery Type

You can generate planets, clouds, mountains, and contours from a single set of parameters. Planet forgeries generate a single planet set against a dark background with stars. Cloud forgeries generate white clouds against a bright blue sky. Mountain forgeries generate a section of a mountain range. Contours are a color representation of the data, rather than forgeries of natural phenomena.

To generate a specific type of forgery, select one of the following menu buttons on the main screen:

- F4 Planet
- F5 Clouds
- F6 Mountain
- F7 Contour

The program generates a low-resolution image in the currently selected image area.

You can use the Options menu to further define the image and then redisplay it in the same image area or in another area. When you find an interesting image, you can render it to the full screen. For rendering information, see "Rendering an Image" on page 166.

Displaying Parameter Settings

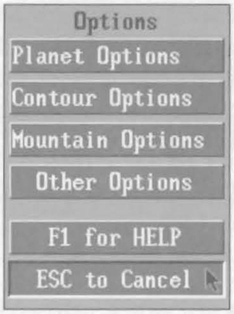
Keyboard shortcut: Alt+1, Alt+2, Alt+3, Alt+4

The Fractal Forgeries main screen displays the parameters for the most recently displayed image. However, you can display the parameter settings for any image in the four image areas and make that image area the current one.

- Right-click in an image box or press Alt+1, Alt+2, Alt+3, or Alt+4 to display the parameters for that image. The highlighted border moves to the image you selected.

Using the Options Menu

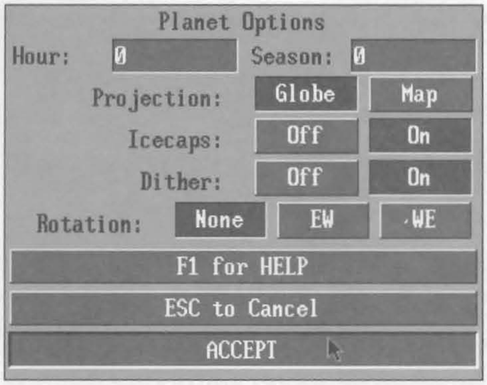
Left-click the F3 Options button at the upper-left corner of the screen or press F3 to display the Options menu.



The Options menu lists the names of the Planet, Contour, Mountain, and Other options submenus.

Planet Options

To display the Planet Options menu, select the Planet Options button on the Options menu.



Planet Options

Hour: 0

Season: 0

Hour

The Hour option lets you change the planet's rotation shadow or "dark side" relative to its sun (or star). You can enter an hour from –24 to 24 to simulate the planet's rotation around its axis. The default setting is 2.

If the hour is set to 0, the sun is directly in front of the planet and no shadow appears on the planet. If the hour is set to 12 (and the season is set to zero) the entire planet is in shadow.

- Enter a number from **–24** to **24** and press ↵ to change the rotation shadow.

Season

The Season option lets you change the planet's dark side by changing the angle of its sun (or star). When the season is set to zero degrees, the sun and the planet's orbit are on the same plane. The default setting is 19.

- Enter a number from **–360** to **360** and press ↵ to change the angle of the sun.

Projection

You can project the image of a planet as a globe (the default) or as a flat, rectangular map that fills the entire image area.

- Select the Globe button to display the planet as a globe.
- Select the Map button to display the planet as a flat, rectangular map.

Note: The Hour and Season options do not affect planets displayed as maps.

Icecaps

Icecaps gives you the option of displaying a planet with or without icecaps at the top and bottom. The default setting is icecaps on.

- Select the On or Off button to turn the display of icecaps on or off.

Dither

The Dither option applys only to full-screen renderings and animations on VGA monitors. *Dithering* paints the image as a series of dots that give the colors a more blended or *halftone* look. Dithered images look very good from a distance. If Dither is off, the image is crisper from close up, but the colors do not blend as well. The default setting is Dither on.

- Select the On or Off button to turn dithering on or off.

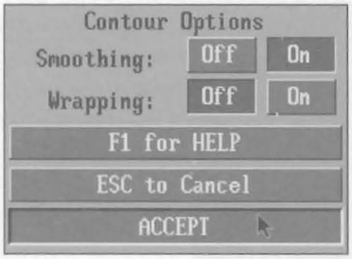
Rotation

Rotation applies only to animation. When rotation is set to EW (east to west) or WE (west to east), the planet rotates from frame to frame and completes a full rotation over the course of the animation. The default setting is none.

- Select the None button to turn off rotation.
- Select the EW button to set the rotation from east to west.
- Select the WE button to set the rotation from west to east.

Contour Options

To use contour options, select the Contour Options button on the Options menu. The Contour Options dialogue box appears.



Contour Options

Smoothing: Off On

Wrapping: Off On

Smoothing

Use the Smoothing option if you want the contour to be painted with curving, irregular shapes. If the Smoothing is turned off, the image is created with squares of separate colors. The default setting is Smoothing on.

- Select the On or Off button to turn Smoothing on or off.

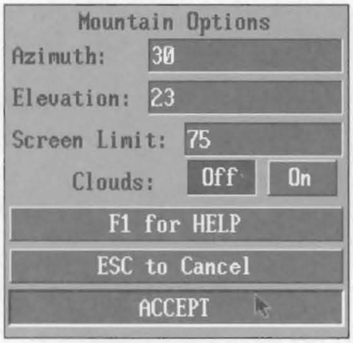
Wrapping

The Wrapping option only applies when Smoothing is on. If wrapping is on, the image is continuous in both the east-west and north-south direction. The default setting is Wrapping off.

- Select the On or Off button to turn Wrapping on or off.

Mountain Options

To use mountain options, select the Mountain Options button on the Options menu. The Mountain Options dialogue box appears.



Mountain Options

Azimuth: 30

Elevation: 23

Screen Limit: 75

Clouds: Off On

Azimuth

The Azimuth option rotates the entire mountain range in a horizontal direction to give you a different view of the landscape. You can change the rotation up to 180 degrees in a negative or positive direction. The default setting is 30 degrees.

- Enter a number from **–180** to **180** and press ⏎ to change the rotation.

Elevation

The Elevation option lets you change the angle of the viewpoint in a vertical direction. The elevation angle can be from zero to 90 degrees. At an elevation of zero, you look head-on at the mountains. At an elevation of 90 degrees, you see a birds-eye view of the mountains (which are represented as a flat map). The default setting is 60 degrees.

- Enter a number from **0** to **90** and press ⏎ to change the elevation.

Screen Limit

Screen Limit determines the percentage of the screen height that is filled by the mountain range. If you set a screen limit of 100 percent, the tallest peak will extend to the top of the screen. The default screen limit is 75.

- Enter a number from **2** to **100** and press ↵ to change the screen limit.

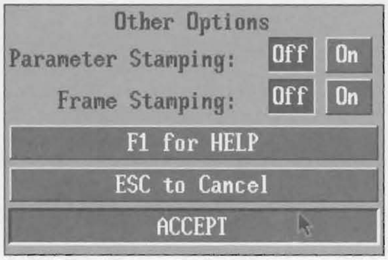
Clouds

You have the option of displaying clouds in the background of mountains on full-screen renderings and in animations. When you render mountains with clouds, the program first paints the clouds on screen, and then makes a second pass to paint the mountains. The default setting is Clouds turned off.

- Select the On or Off button to turn the display of clouds on or off.

Other Options

There are two additional options that control whether information is displayed on renderings and animations. To display the Other Options menu, select the Other Options button on the Options menu.



Other Options

Parameter Stamping: Off On

Frame Stamping: Off On

Parameter Stamping

Parameter stamping gives you the option of displaying parameter values on a full-screen rendering. The default setting is Parameter Stamping turned off.

- Select the On or Off button to turn the display of parameters on or off.

Frame Stamping

Frame stamping gives you the option of displaying animation frame numbers on screen. The default setting is Frame Stamping turned off.

- Select the On or Off button to turn the display of frame numbers on or off.

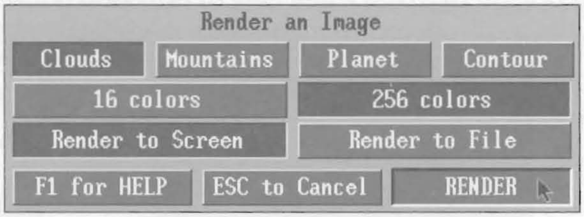
Note: If the Frame Stamping option is on when you create an animation, the frame numbers appear when you view the animation, even if you turn Frame Stamping off at that time.

Rendering an Image

Keyboard shortcut: R to bypass Render an Image dialogue box

After you set all the parameters and select the appropriate options for the type of image you want to create, you can render the image on a full screen or to a file. On VGA monitors, you can render the image in 256 colors.

To render an image, left-click the F8 Render button on the Main menu or press F8 . The Render an Image dialogue box appears.



In this dialogue box, you specify the type of image you want to create, the number of colors you want to use, and whether you want to render to the screen or to a file.

Selecting the Image Type

To specify the type of image you want to render, select one of the following buttons:

- Clouds
- Mountains
- Planet
- Contour

Selecting the Number of Screen Colors

If you have a VGA monitor, you have the choice of rendering the image in 16 or 256 colors.

- Select the 16 colors or 256 colors button to choose the number of colors for the rendering.

Rendering to the Screen or a File

You can render to the full screen or you can save the rendering in a file.

To render to the screen, follow these steps:

1. Select the Render to Screen button.
2. Select the RENDER button to render the image with the selected render options. The program displays the image on the full screen.

To save the rendering in a file, follow these steps:

1. Select the Render to File button.
2. Select the RENDER button to accept the rendering options. A filename dialogue box appears.
3. Enter a name for the rendering file and select the OK button. The program saves the rendering to that file and displays the image on the full screen.

Exiting the Full-Screen Rendering

To exit the full-screen rendering and return to the opening screen

- Click either mouse button or press any key.

Animating an Image

If you have a VGA monitor, you can animate a forgery. To create an animation, you specify starting and ending parameters and tell the program how many frames you want to distribute the changes over. For example, you can create an animation of a planet forgery that starts with the hour set to zero and ends with the hour set to 24. When you play back the animation, you'll see the rotation shadow move across the planet.

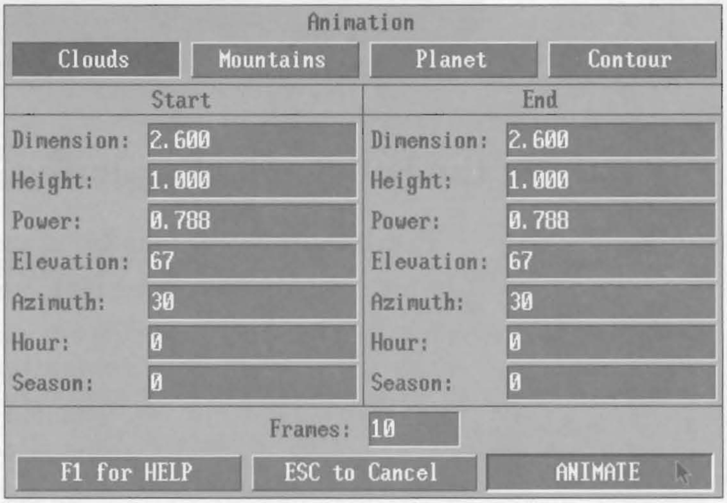
When you create an animation, the program saves it in an animation file with a *.fli* extension. You can play back the Animation in the Fractal Forgeries program or you can use the animation file in Autodesk Animator™.

Creating an Animation

Keyboard shortcut: A to display the Animation dialogue box

To create an Animation, follow these steps:

1. Select Animate from the F2 File menu. The Animation dialogue box appears.



	Start	End
Dimension:	2.600	2.600
Height:	1.000	1.000
Power:	0.788	0.788
Elevation:	67	67
Azimuth:	30	30
Hour:	0	0
Season:	0	0

Frames: 10

Note: If you have set starting and ending animation parameters on the main screen as described in "Setting Animation Parameters" on page 157, those parameters are shown in the Animation dialogue box.

2. Select one of the following buttons for the type of image you want to animate:
   - Clouds
   - Mountains
   - Planet
   - Contour
3. Make any changes you want to the Start and End parameter fields for the animation sequence.

   Note: Dimension, Height, and Power are described in "Changing Forgery Parameters" on page 155. Elevation and Azimuth are described in "Mountain Options" on page 164. Hour and Season are described in "Planet Options" on page 161.
4. Enter the number of frames for the animation in the Frames field or use the default setting of 10 frames. The acceptable range for the number of frames is 1 to 999.
5. Select the ANIMATE button to create the animation file. The Make FLI dialogue box appears.

Making an Animation File

When you select the Animate button in the Animation dialogue box, the program displays another dialogue box requesting a file name for the animation.

To name the animation file, follow these steps:

1. Enter an animation filename and press ⏎. (The program automatically adds the *.fli* extension to your filename.)
2. Press ⏎ or select the OK button to start generating the animation.

The program generates each frame of the animation and displays them on screen. When all frames have been generated, you return to the main screen in the program.

Note: If your computer runs out of memory while generating the animation, you can do one or both of the following and try again: 1.) Increase the amount of memory available to CHAOS (for example, remove any unnecessary TSRs); 2.) Reduce the Mesh Size parameter.

Viewing an Animation

Keyboard shortcut: V

To view an animation, follow these steps:

1. Select View Animation from the F2 File menu. A dialogue box appears requesting the name of the *.fli* (animation) file you want to view.
2. Enter the animation filename and press ↵ or double-click a filename in the list box on the left.
3. Select the OK button or press ↵ to play back the animation. The program displays the animation over and over again.

To stop the animation, follow these steps:

1. Click either mouse button or press any key to return to the View Animation dialogue box.
2. At the View Animation dialogue box, press Esc to return to the main program screen.

Saving and Loading Files

You can use the options on the File menu to save and reload parameter files, *.gif* files, animation files (*.fli* files), and *.dxf* files. For information on saving and loading parameters and *.gif* files, see "Saving and Loading Files" on page 209. For information on saving and viewing animation files, see "Animating an Image" on page 168.

Saving DXF Files

Keyboard shortcut: X

DXF (Drawing Interchange Format) is a file format originally developed by Autodesk as an exchange format for AutoCAD®.

DXF file compatibility has since been incorporated into many drawing and desktop publishing programs.

To create a DXF file using the current parameters, follow these steps:

1. Select the Save as DXF button on the F2 File menu. A dialogue box appears prompting you for a filename.
2. Enter a filename and press ⏎. The program automatically adds the *.dxf* extension to your filename.
3. Press ⏎ or select the OK button to save the *.dxf* file.

You can import the *.dxf* file into many desktop publishing and graphics programs including AutoCAD and 3D Studio™.

The Mathematics of the Program

The Fractal Forgeries program uses a delicately tuned mixture of order and randomness to generate shapes that resemble clouds, mountains, and planetary landscapes. The random element of each image is involved in the way we pick certain coefficients, and the orderly element is the way in which the coefficients are used to build up a series of curves that are summed to produce the eventual pattern. Although the forms of naturally occurring clouds, mountains, and planets are probably not determined by algorithms as simple as ours, naturally occurring shapes lie at the same kind of chaotic interface between randomness and order.

Our program works by a technique known as the "spectral synthesis of a Fourier sum." The basic notion is that an arbitrarily complicated curve $y = F(x)$ can be gotten by taking a weighted sum of sine waves of various frequencies. The sine function is the familiar periodic curve that smoothly wiggles up and down as you move out along the x-axis. Sin(2*x) wiggles twice as fast as sin x, sin(3*x) has triple the frequency, and so on. (Our Fourier sums actually use terms of the form cos(n*x) as well, but we'll leave them out here to make the exposition simpler, since this only affects the phase of the waves.) In one dimension, a general Fourier sum using n different frequencies looks like this:

$$F(x) = a_0 + a_1 * \sin(x) + a_2 * \sin(2*x) + \ldots + a_n * \sin(n*x).$$

Fourier sums also work in two dimensions, where a surface can be thought of as the graph of a height function $z = G(x,y)$. The building blocks used here are quilt-like surfaces defined by formulas of the form sin(x) * sin(y), sin(x) * sin(2*y), sin(2*x) * sin(y), and so on. These Fourier sums also use terms of the form cos (n*x), but, again, we'll leave them out to make the exposition simpler. In two dimensions, a Fourier sum using n*n frequencies looks something like this:

$$G(x,y) = a_{00} + a_{01} * \sin(x) + a_{10} * \sin(y) + a_{11} * \sin(x) * \sin(y) + a_{12} * \sin(x) * \sin(2*y) + a_{21} * \sin(2*x) * \sin(y) + a_{22} * \sin(2*x) * \sin(2*y)+...+ a_{nn} * \sin(n*x) * \sin(n*y).$$

If we know in advance exactly what our surface G is supposed to look like, then it is possible (though exceedingly complex) to calculate the best choices for the Fourier coefficients a_{ij}.

Conversely, it is also possible to generate entirely new surfaces by picking the a_{ij} by some more or less random process, summing up the terms, and drawing a picture of the surface that results.

When our Fractal Forgeries program is used to draw a Mountain, this is exactly what is going on. The Cloud, Planet, and Contour representations are really just different ways of presenting the same calculated data, as will be explained in more detail below. For now, however, it is easiest to think of Fractal Forgeries as a program that generates 3D views of surfaces.

When it creates a surface, Fractal Forgeries first picks some reasonable values for the a_{ij}. As these coefficients determine the weights assigned to the different possible frequency components, they can be thought of as influencing the "spectrum" of frequencies that the surface will use. The process of picking the a_{ij} is known as "Spectral Synthesis." Once the a_{ij} are set, the program uses the G(x,y) formula from above to evaluate G(x,y) at some (though not all) of the possible (x,y) positions. The calculation of the Fourier sums is called an "Inverse Fourier Transform."

How exactly do we perform our spectral synthesis? If we were to pick the a_{ij} completely at random, we'd get surfaces that looked quite unnatural. Instead, we specify for each combination of i and j what the expected value of a_{ij} ought to be, and then try and pick some random value that is near this expectation. The surface G's bumpiness or "fractality" depends quite sensitively on the way that the expected value of a_{ij} varies with the size of i and j. Keep in mind that a term $a_{ij}*\sin(i*x)*\sin(j*y)$ is able to contribute more bumpiness to the surface if both i and j are large—large i and j mean a rapidly oscillating surface—also keep in mind that a_{ij} determines the size, or amplitude, of these oscillations. If we request that the expected value of a_{ij} be quite small for large values of i and j, then the Fourier sum surface will be smooth. If, on the other hand, the expected values of the a_{ij} remain reasonably large as i and j grow, then the Fourier sum surface will have more ups and downs.

A perfectly smooth surface has dimension 2. From the standpoint of fractal mathematics, the more bumpy a surface is, the higher is its fractal dimension. If a surface were endlessly spiky—like the surface made up of the combined surfaces of the bristles of a brush, say—then it would effectively fill up all of a three-dimensional region. As a surface gets more and more bumpy and fractured, its fractal dimension gets closer and closer to 3.

When the user sets the Fractal Dimension parameter between 2 and 3, he or she is essentially specifying how rapidly the average sizes of the a_{ij} parameters should fall off with increasing values of i and j. Keep in mind, however, that since we only use a relatively small number of terms in our Fourier sums, the so-called "random fractals" that our program generates do not have anything like the detail of the "deterministic fractals" generated by The Chaos Game and by the Mandelbrot program.

What about the other parameters: Height, Power Factor, Mesh Size, and Terms?

The Mesh Size parameter controls two quite disparate things. First of all, it controls the size of the maximum frequency number N, which is used in the Fourier sum. The Fourier sum potentially allows all terms of the form $a_{ij} * \sin(i*x) * \sin(j*y)$ with i and j both ≤Mesh Size. The Mesh Size parameter also happens to be used for a wholly unrelated parameter: as a measure of the number M of points (x,y) for which the full Fourier sum is actually calculated. We use a grid with Mesh Size points on each edge. You can see the mesh quite plainly if you draw a Contour image after using the Contour Options menu to turn the Smoothing feature off. Another way to see the mesh is to draw a Mountain image after using the Mountain Options menu to change the Azimuth to 90. The "mountain heights" are calculated at each of the mesh points, and eventually the values for the points in between the mesh points are interpolated by filling in the spaces between adjacent triples of mesh points with triangular pieces of plane.

The reason that we take the highest frequency number N and the mesh edge length M to be the same number is that it would effectively be a waste of computation to include frequencies which oscillate much more rapidly than the steps of our mesh. But we do allow the user to hand-set the maximum frequency to be lower than the number of mesh points.

The Terms parameter controls how many of the possible Fourier sum terms we actually use. If Mesh Size is M, and Terms is N, it really only makes sense to have $N \le M$. If you select an $N > M$, the program will actually behave as if you had selected $N = M$. The effect of reducing N is to zero out more and more of the Fourier terms $a_{ij} * \sin(i*x) * \sin(j*y)$, making the surface smoother and smoother. This is slightly different from the effect of setting a low fractal dimension to make the a_{ij} be very small. If, for instance, we have a high fractal dimension but only use a few terms, then we'll get reasonably large a_{ij} out to some point—at which they switch to being zero. This kind of combination can make pleasingly hilly terrain.

As we compute the value of the Fourier sum G(x,y) for each mesh point, we keep track of their sizes so as to determine the largest value attained, which we can call MaxZ. Once all the Fourier sums are calculated, we set all the G(x,y) which came out negative to 0, and we divide all the positive G(x,y) by MaxZ. The result of this is that all the G(x,y) now lie between 0 and 1. The next step of our landscape synthesis is known as "Power Scaling," and is determined by the value of a user-specified Power Factor, which lies between 0 and 2. What we do is to raise each of the G(x,y) to a power E, which is based on the Power Factor.

Keep in mind that $0 \le G(x,y) \le 1$. If we square a number z that is between 0 and 1, say 0.4, then we get a smaller number. $0.4^2 = 0.16 < 0.4$. In general, if E is any exponent greater than 1 and z is any number between 0 and 1, then z^E is less than z. Conversely, if we were to take the square root of, say, 0.49, we would get a larger number, in this case 0.7. Recall that taking a square root of a number is the same thing as raising that number to the power 0.5. In general, if E is any exponent between 0 and 1, and z is a number between 0 and 1, then z^E is greater than z. If P is our power factor, then we actually take E to be $e^{(P-1)}$, so that the effective range of E is about 0.28 to 1 to 2.6, as the parameter P ranges from 0 to 1 to 2. The net effect is that if the power factor is less than 1, then it pushes all but the highest points up—creating things like mesas—and if the power factor is greater than one, it drags all but the highest points down—making spires.

At this point, all the G(x,y) still range from 0 to 1. If we have selected Cloud, then we bail out here, and color all the G(x,y) with values less than 0.5 blue, and scale all the values between 0.5 and 1.0 into increasingly saturated shades of white. If you find that your clouds are coming out with too much blue, set the Power

Factor to a *low* value so as to make the G(x,y) get bigger. Remember that for G(x,y) to show as white it has to be bigger than 0.5. If we've selected Contour, we also bail out here, using an unsmooth range of colors to show the steps from 0 to 1.

If we're doing Mountain or Planet, then we do one more operation, multiplying the G(x,y) by the user-specified Height parameter.

To show a Planet, we use a spectrum of colors that makes points less than 0.5 look like they're under water. To show a Mountain, we calculate all the heights and rescale one last time to make sure the highest peak fits on screen. Before we actually draw a mountain range on screen, we draw a virtual image of it in memory, and eliminate all the pieces that are behind other pieces. This last process is called "Z-buffering."

The wiggly line in the window at the top of the menu is a fast preview of what the more complicated images are expected to look like. The line might be thought of as a diagonal cross-section of one of the Mountain images, taken from corner to corner.

The randomizer algorithm uses a certain seed number to start with. If you reset the seed to the same value twice, you'll get the same pseudorandom number sequence, resulting in exactly the same "random" fractal forgery. If you like, you can use something like your birthday for the random seed. Just make sure to order the digits of your birthday in such a way that the leading digit is less than 3, because the largest "seed" that our randomizer likes to eat is about 32,000.

All of our "random" fractals are based on sine waves, and sine waves are periodic. They repeat themselves. This fact is evident if you look at one of our images closely—the left edge always matches the right edge, and the top matches the bottom. If you could print out lots of them, you could set them edge to edge like Escher tiles. This periodicity comes in handy when we wrap the images around a spherical planet and let the planet rotate.

Chapter 7

Toy Universes

In our world, complexity flourishes . . . Somehow, after all, as the universe ebbs toward its final equilibrium in the featureless heat bath of maximum entropy, it manages to create interesting structures . . . the changing degree of form and formlessness in the creation of amino acids, of microorganisms, of self-reproducing plants and animals, of complex information systems like the brain. (Chaos: Making a New Science, *page 308.)*

Imagine a forest where every tree tries to grow higher than its neighbors. Say that every spring, each tree takes account of the average height of its nearest neighbors and then grows ten feet higher than that average over the coming year. Suppose also that once a tree gets to a maximum height of sixty feet, it stays at that height for a year, and then dies, leaving a blank space. Finally, suppose that a blank space will sprout a new tree provided that there are living trees nearby.

What kind of growth pattern would such a forest display? If the forest started out from a single tree, one would expect orderly circles of growth and decay, endlessly radiating out from the position of the first tree. But what if the forest were started with a random scattering of seeds, some bunched together, and some far apart?

As it turns out, even a randomly started forest will develop orderly waves of growth. Rather than being perfect circles, the waves will be curved arcs that look like spirals at either end. Several kinds of chaotic systems generate these shapes, which resemble the capitals of Ionic columns. The Soviet chemists B.P. Beluzov and A.M. Zhabotinsky first observed these shapes in petri-dish experiments in the late 1950s. To set off the so-called *BZ reaction,* they would mix some reactive chemicals with an indicator fluid whose color changes made visible the solution's local acidity at each point. To set the BZ reaction in motion, they would sprinkle in a catalyzing dust of palladium crystals. Each crystal acts like a "seed" in the forest example described above. Though the pattern is thoroughly random to start with, before long the characteristic double-spiral scrolls, now known as *BZ-scrolls,* always appear.

Building on the work of Beluzov and Zhabotinsky, the biologist Arthur T. Winfree ran computer simulations to show that BZ-scrolls can occur in three-dimensional reactive media, and conjectured that the wave-fronts of excitation that keep our hearts beating can be fruitfully thought of as BZ-scrolls.

The BZ-scrolls shown in the Toy Universes program are examples of a kind of computer simulation known as *cellular automata.* Cellular automata were invented by John von Neumann and Stanislaw Ulam in the late 1940s as a way of modeling a kind of artificial life. Von Neumann's goal was to show that there can be man-made patterns of information that reproduce themselves, just as do all the truly living creatures around us. Initially, he tried to couch his argument in terms of robots that assemble other robots, but the details of this approach grew too unwieldy. Ulam, a mathematician who helped invent the hydrogen bomb, gave von Neumann the idea of modeling a computer creature as a certain pattern in a grid of square cells. By the mid-1950s, von Neumann was able to prove that there are cellular automata worlds in which some of the patterns do indeed build copies of themselves. Presciently enough, his proof hinges on the idea of having his cellular automata creatures carry copies of their blueprints; years later, Watson and Crick would establish that real living creatures carry blueprints in the form of DNA.

A very simple cellular automaton rule known as the Game of Life, created by the mathematician John Horton Conway, became popular among computer hackers in the 1970s. In Life, each cell looks at its eight nearest neighbors and decides at each generation whether to turn itself on or off.

The four families of cellular automata shown in Toy Universes can use many more than Life's two states. Each of their initial parameter settings is tuned to produce one of the patterns that cellular automata seem to do best: BZ-scrolls. It is striking to see how this apparently universal shape emerges from the running of each of the quite different rules. In the Hodge rule, in particular, the scrolls have a hauntingly organic look to them.

In some chaotic computer simulations, we see very simple equations generating complex fractal patterns. In the toy universes of cellular automata, we see that the process can run the other way. By pressing F5, you can seed any of these simulations with a completely disorderly start condition, and then watch how the disorderly start slowly evolves into the chaotic order of the BZ-scrolls.

The program seems to suggest that chaos exists as a kind of interface between order and disorder.

Things To Try

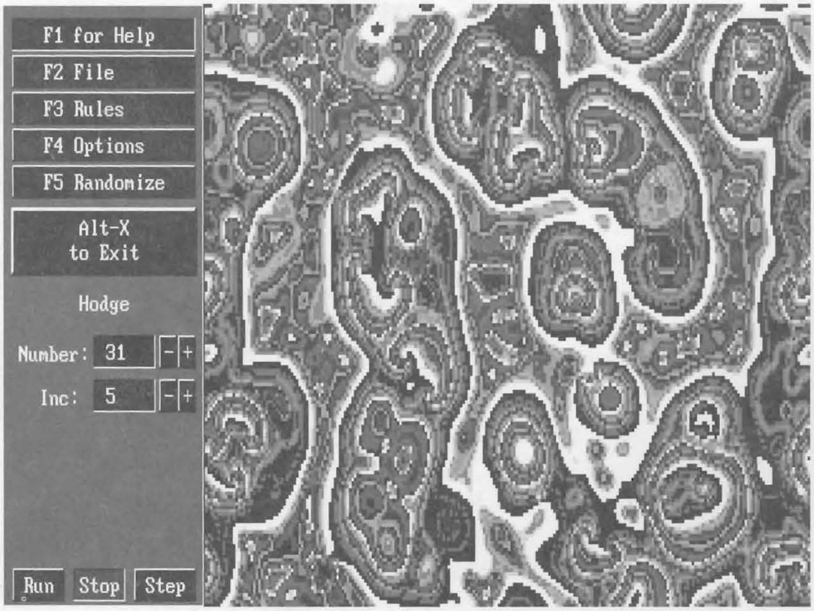
Select the F3 Rules button to cycle through four different kinds of cellular automata. When you select a new rule, it is a good idea to re-seed the screen by pressing F5 to randomize it.

Try pressing Alt + Z to get a zoom box and then left-click the mouse or press the Ins key to zoom in. You can speed up the zoomed-in toy universes by turning on the Fast option on the Options menu.

You can also zoom out to a slow, but very high resolution mode by right-clicking or pressing Del. It is interesting to let the Eat rule or the Tube rule run for an hour or so in this high resolution mode.

Starting the Program

When you start the program, you see this screen:



Hodge

Number: 31 - +

Inc: 5 - +

Run, Stop, and Step buttons

The main image area of the screen shows a *cellular automaton* (or toy universe). Cellular automata are self-generating, ever-growing computer graphics with cells whose colors change according to different rules.

At the bottom of the menu area, you see the Run, Stop, and Step buttons, which make each toy universe start moving, stop moving, or move one step at a time.

Exploring the Program

Have fun exploring the cellular automata in this program before you work with each feature in detail. You learn more about the features of Toy Universes later in this chapter.

Begin by cycling through this program's four different rules: Hodge, Eat, NLUKY, and Tube. Here's how to display and run the rules.

1. Watch the Hodge cellular automaton change for a while after you first start the program.
2. Select the Stop button or press the spacebar to stop the growth of the cellular automaton.
3. Left-click the F3 Rules button or press F3 to display the next rule.
4. Select the Run button or press Shift and the spacebar to start the cellular automaton's growth, and then immediately left-click the F5 Randomize button or press F5 to randomize the new rule. Watch the new rule run for a while.
5. Cycle through the remaining two rules—NLUKY and Tube—by repeating steps 2–4.
6. Finally, press the spacebar to stop the Tube rule and press F3 to display the Hodge rule again.

Note: After you change to each new rule and start it running, always randomize the cellular automaton. Randomizing starts cell growth from scratch and helps keep it going.

You might enjoy experimenting with colors in the program as the toy universes change in the image area.

1. Display the Eat rule, start it running, and randomize it.

2. Press Alt+Y to start colors cycling continuously while the rule grows. The cycling colors bring out different shapes as the cellular automaton changes in the image area.
3. As the colors cycle, press Alt+2 to overlay a diamond shape in the cellular automaton. The color-cycling cellular automaton "eats" toward the center of the diamond that the program overlays in the rule.
4. For a shimmering effect while the colors are still cycling, press the Stop button or the spacebar.
5. Press Alt+Y two more times to turn the continuous color cycling feature off. Press Alt+D to return to the default palette.

Zooming In on a Cellular Automaton

You can zoom in for a closer look or zoom out to see a wider area. In Toy Universes, you can zoom to four different resolutions—low, medium, high, and very high. The program starts in high resolution. (If CHAOS detects that there is not enough memory in your computer the first time you try to zoom out to very high resolution, the program displays a warning message. After that, the program ignores any attempts to zoom out to very high resolution until it detects enough memory to do so.)

Note: If the cellular automaton is stopped when you zoom, the image area will turn black. If this happens, select the Run button or press Shift and the spacebar to start the cellular automaton at the new resolution you just zoomed to.

To zoom with the mouse

1. With a cellular automaton running and the cursor anywhere in the image area, left-click to zoom in. (Since you zoom on the center of the image area, cursor location within the image area is not important.)
2. With the cursor anywhere in the image area, right-click to zoom out.

To zoom with the keyboard

1. With a cellular automaton running and the cursor anywhere, press Ins.
2. With the cursor anywhere, press Del to zoom out.

You can also use a zoom box that lets you choose the center of the image you want to zoom on. You can zoom with the zoom box only at medium or high resolution. You cannot resize the zoom box in Toy Universes.

1. Press (Alt)+(Z) to display the zoom box.
2. Move the zoom box to the location you want to center your zoom on.
3. Left-click or press (Ins) to zoom in on the specific area.
4. With the zoom box displayed, you can also zoom out by right-clicking or pressing (Del).
5. If you change your mind while you're working with the zoom box, you can press (Esc) to cancel the zoom box.

Note: To move the zoom box around the image area with the keyboard, use the arrow keys, as well as (PgUp), (PgDn), (Home), and (End).

Examining Growth in Steps

You can examine a cellular automaton's growth one step at a time. When you look at the growth in steps, you get a better idea of the color changes that are happening.

1. Select the Step button or press the spacebar to stop the cellular automaton's growth in the image area.
2. Select the Step button or press the spacebar repeatedly to examine the growth one step at a time.
3. When you're through watching the growth in steps, select the Run button or press (Shift) and the spacebar to go back to free-running growth.

Note: If you randomize while you are examining cell growth one step at a time, you must select the Step button or press the spacebar once for randomization to take effect.

Understanding Cellular Automata

Though the patterns look complicated, the logic of the toy universes is simple. Each type of cellular automaton is made up of cells obeying a single *rule*. A rule changes the color of each cell.

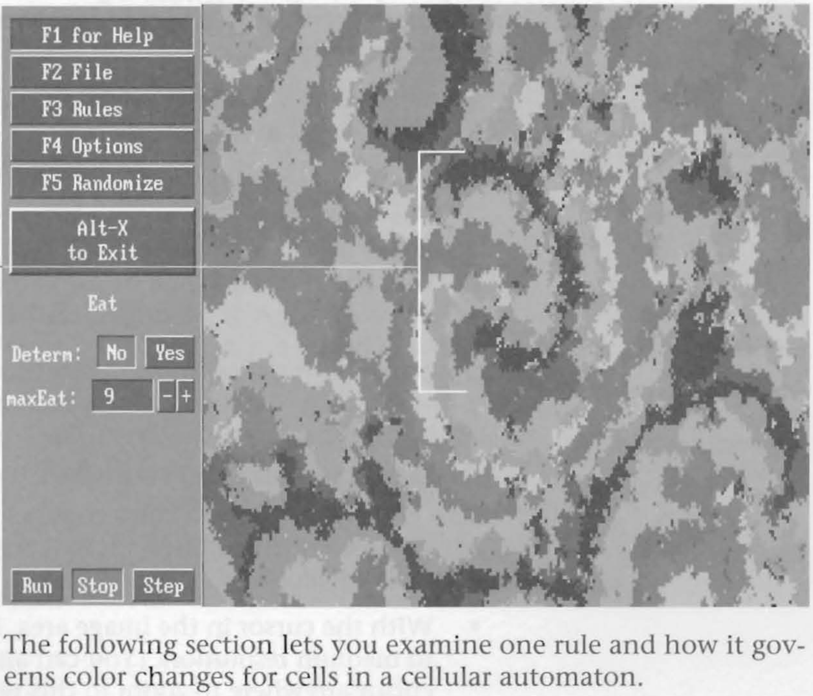
A cell's *neighbors* are the eight cells that surround and touch it on all sides. In this program, a cell makes its next color change on the basis of the present colors of itself and its neighbors.

Three things characterize cellular automata:

- They are *parallel*. All cells compute their new colors at the same time.
- They are *homogeneous*. Each cell obeys the same rule for color changes.
- They are *local*. Each cell looks only at its closest neighbors in deciding what color to change to.

In addition, many cellular automata evolve into images with a common theme. After a cellular automaton runs for a while, you often see a shape like the double-spiraled scroll at the top of an Ionic column. People who work with cellular automata call these shapes *BZ-scrolls*, after two Soviet chemists, Beluzov and Zhabotinsky, who discovered the shape in chemical reactions. (You can see photographs of these chemical reactions in James Gleick's *Chaos: Making a New Science,* page 287.)

Here is an example of the Eat cellular automaton with BZ-scrolls:



Determ: No Yes

maxEat: 9 - +

BZ-scroll

The following section lets you examine one rule and how it governs color changes for cells in a cellular automaton.

Understanding the Hodge Rule

In the Hodge rule, each cell sums the values of its eight neighbors, averages that total, and adds the value stored in the Increment parameter. The result is the cell's new *color value* (the number assigned to a particular color). When a cell's new color value exceeds the value stored in the Number parameter, the cell stays at the color value of the Number parameter for one generation. Then the cell drops back to zero, which is black. The cell changes color value again when the sum of its neighbors exceeds a certain threshold value. For details of the calculations in the Hodge rule, see "The Mathematics of the Program" on page 202.

To help you understand cellular automata rules, you can examine the Hodge rule at work and the color changes in one cell.

Important: Be sure you are working with the default Hodge palette as you do the exercises in this section. If you are unsure what palette you have for the Hodge rule, exit the program and reenter, or, with the Hodge label displayed in the menu area, press Alt + D to load the default palette.

Start by simplifying the Hodge rule's color changes. Make the following changes in the Hodge rule's parameter buttons and then start the rule running:

1. Select the F3 Rules button or press F3 until you see the Hodge label in the menu area.
2. Enter **6** in the Number field, and press ↵.
3. Enter **2** in the Inc field, and press ↵.
4. Select the Run button or press Shift and the spacebar to start the rule, and then immediately left-click the F5 Randomize button or press F5 to randomize the new rule.

When you set up these parameters and start the rule working, you see cells of just a few colors (orange, brown, dark red, blue, and green), in the default palette. The color areas grow and merge on screen.

You can get a closer view of color changes in individual cells by zooming to medium resolution. At this resolution, there are grid lines around each cell.

- With the cursor in the image area, left-click once to zoom to medium resolution. (You can also press Ins with the cursor anywhere to zoom to this resolution.) Then immediately select the F5 Randomize button or press (F5) to randomize the cellular automaton again.

Even after you zoom to this resolution, you probably find that the color changes happen so fast that you cannot comprehend all the changes.

You can get a clear view of what is happening to cell colors by watching the rule run in single steps. At the same time, you can find out what the color values are for the five colors in this setup of the Hodge rule.

1. Again, select the F5 Randomize button or press (F5) to randomize the rule, and then select the Step button or press the spacebar right away. Cell growth pauses.
2. After that, select the Step button or press the spacebar repeatedly and note each different cell color as it comes in. As each new color comes in, assign 0 to black, 1 to the first color that comes in, 2 to the second color, and so on, through 5. The color values in this default palette are

   0 = black
   1 = orange
   2 = brown
   3 = dark red
   4 = blue
   5 = green

Now you're ready to focus on one cell and watch how the Hodge rule changes the cell's colors. Follow these steps:

1. With cell growth still paused in the single-step mode, choose a non-black, non-green cell.
2. Sum the color values of the cell's eight neighbors.
3. Divide by 8 and round down to the nearest whole number.
4. Add 2 (the value stored in the Inc parameter).

The result of this computation should be the color value for the cell the next time you press the spacebar.

For example, look at the following configuration of color values for the neighbors of one cell:

1	*2*	*3*
2	*cell*	*4*
1	*2*	*3*

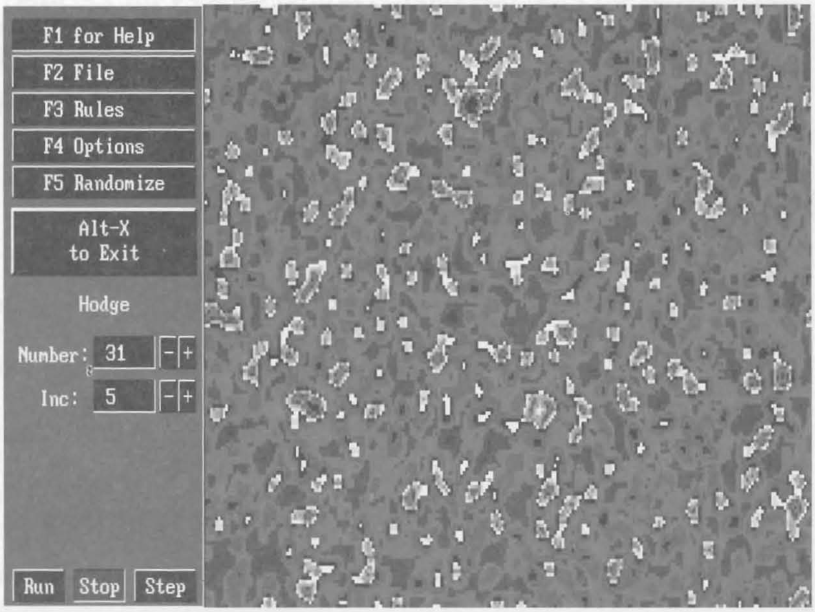
To understand how the Hodge rule works, perform the computation in the following steps:

1. First, sum the color values of the cell's eight neighbors. The result is 18.
2. Then, divide by 8 and round down to a whole number. The result is 2.
3. Finally, add 2, which is the value stored in the Inc parameter. The result—4—gives you the cell's next color value. Looking back at the color values you observed for all five colors in the cellular automaton, you see that 4 = blue. So, the next time you press the spacebar, the cell in this illustration would change to blue.

Changing the Hodge Parameters

If Hodge is not already displayed, left-click the F3 Rules button or press F3 until the program displays the Hodge parameters in the middle of the menu area:

Hodge Rule image



Number: 31

Inc: 5

Hodge parameters

Use the Hodge parameters to change the Hodge rule. In general, the best BZ-scrolls appear if the Number parameter is about six times the size of Inc. If Number is very large relative to Inc, smoothly averaged contour lines emerge.

You can increase or decrease both parameters the (+) or (−) buttons next to each entry field.

Suggestion: Try using a large Number parameter and a small Inc parameter and turning off screen wrap by pressing (W). If you set up the rule this way and let it run for a few minutes, you see patterns like those in hooked rugs.

Number

Keyboard shortcut: (Shift)+(N) to increase; (N) to decrease

The Number parameter sets the maximum number of color states that cells can change to in the Hodge rule's computation. The default number is 31.

- Enter a value between **1** and **255** and press (↵).

Inc (or Increment)

Keyboard shortcut: Shift + I to increase; I to decrease

The Inc parameter sets the value to be added in each computation for the Hodge rule. The default increment is 5.

- Enter a value between **0** and **255** and press ↵.

Understanding the Eat Rule

In the Eat rule, each cell changes color based on whether its neighbors' color values are cyclically one unit higher than the cell's own color value. Cells can change from color values 0 through one less than whatever value you set in the MaxEat parameter. Thus if MaxEat is 3, the cells can become color values 0, 1, or 2. If you think of these values as being in a cycle, with MaxEat set at 3, then 0 can be eaten by 1, 1 can be eaten by 2, 2 can be eaten by 0, and so on. In general, (MaxEat – 1) can be eaten by 0.

A cell looks at the color value of either one neighbor or four neighbors, depending on whether you are running the rule with deterministic mode set to No or Yes.

With deterministic mode set to No, the rule has a random component. Each cell randomly looks at the color value of just one of its eight neighbors. If the neighbor's color value is cyclically one unit higher than the cell's, the neighbor's color value "eats" (or replaces) the cell's. If the color value of the neighbor is any value other than one unit cyclically higher, the cell stays the same color.

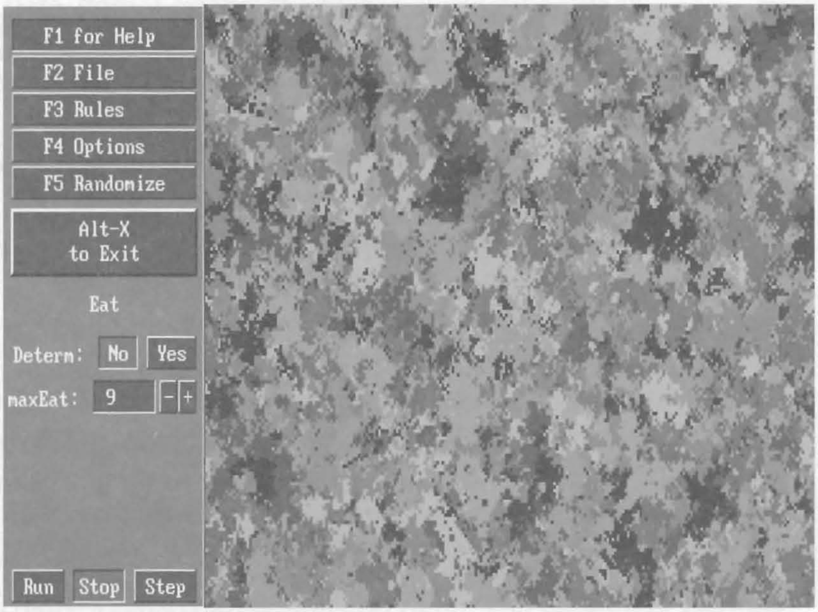
With deterministic mode set to Yes, each cell looks at the color values of four of its neighbors—the neighboring cells directly to either side of the cell, and directly above and below it. If any one of their color values is one unit higher than the cell's, that neighbor's color value "eats" the cell's, just as when the deterministic mode is set to No. If the color values of the four neighboring cells are all values other than one unit higher, the cell stays the same color.

For details on the calculations in the Eat rule, see "The Mathematics of the Program" on page 202.

Changing the Eat Parameter and Mode

Left-click the F3 Rules button or press F3 until the program displays the Eat parameter and mode in the middle of the menu area:

Eat Rule image



Eat Mode — Determ: No Yes

Eat Parameter — maxEat: 9

Use the Eat parameter and mode to change the Eat rule. In general, be sure to randomize the rule by selecting the F5 Randomize button or pressing F5 whenever you increase the MaxEat parameter. Increasing the MaxEat parameter above 20 makes it difficult for the rule to get started. The Eat rule reaches equilibrium much faster with deterministic mode set to Yes. If you then set deterministic mode to No, the rule looks fuzzier and more organic. Eventually, BZ-scrolls appear.

Suggestion: A good way to start the Eat rule is to overlay a shape. Press Alt+1 to overlay a square, Alt+2 to overlay a diamond, Alt+3 to overlay a circle, or Alt+4 to overlay the Autodesk logo. You might also enjoy zooming out to very high resolution and letting the cells eat outward to a blank background.

Determ (or Deterministic)

Keyboard shortcut: D

The Determ mode lets you run the Eat rule with or without a random component. When you set this mode to No, a cell randomly looks at just one neighbor in making its next color change. When you set the mode to Yes, a cell looks at neighboring cells to either side of it and directly above and below it in making its next color change. The default condition for the Eat rule is Determ mode set to No.

- Select the No or Yes button to run the Eat rule with or without a random component.

MaxEat

Keyboard shortcut: Shift + E to increase; E to decrease

The MaxEat parameter sets the maximum number of color states that cells can change to in the Eat rule's computation. The default value is 9.

- Enter a value between **1** and **255** and press ↵. You can also increase or decrease MaxEat with the + or − buttons next to the entry field.

Understanding the NLUKY Rule

In the NLUKY rule, a cell can be in one of three kinds of color states: the state for *off,* the state for *on,* or in one of a range of *resting states.* The value you set for the N parameter determines the number of resting colors in the NLUKY rule.

The values you set in the L and U parameters determine when a cell goes from off to on, and the values you set in the K and Y parameters determine when a cell goes from on to a resting state. To make use of these parameters, each cell counts the number of its eight nearest neighbors that are in the on state. This quantity, called the *stimulus,* can range between 0 and 8.

If a cell is off and the stimulus from its neighbors lies between L and U, then the cell turns on. Otherwise, the cell stays off. (If L is not greater than U, stimulus being between L and U means that L is less than or equal to stimulus, and stimulus is less than or equal to U.)

If a cell is on and the stimulus from its neighbors lies between K and Y, then the cell remains on. Otherwise, the cell will move into the lowest resting state. (If K is not greater than Y, stimulus being between K and Y means that K is less than or equal to stimulus, and stimulus is less than or equal to Y.)

If a cell is in a resting state, it always moves into the next resting state, until it reaches the highest resting state. At that point, it moves back down into the off state, and the cycle restarts.

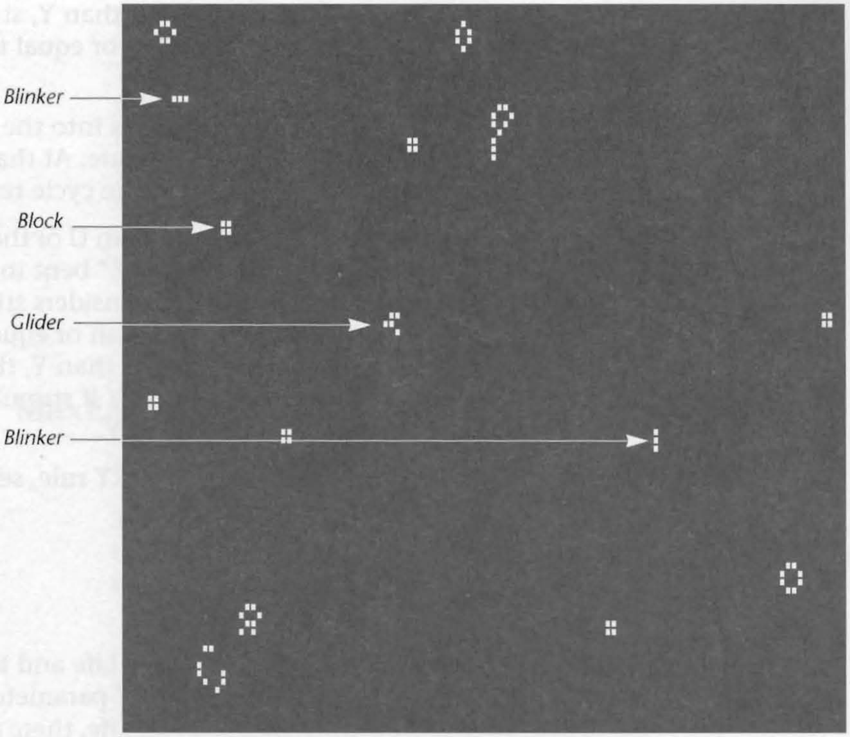
It is also possible that L might be greater than U or that K might be greater than Y. If so, the stimulus is "cyclic," bent into a circle. Thus if L is greater than U, the program considers stimulus to be "between" L and U if stimulus is greater than or equal to L or less than or equal to U. Likewise, if K is greater than Y, the program considers stimulus to be "between" K and Y if stimulus is greater than or equal to K or less than or equal to Y.

For details on the calculations in the NLUKY rule, see "The Mathematics of the Program" on page 202.

The Life Version of the NLUKY Rule

The NLUKY rule has two famous versions: Life and Brain. In the Life version of the rule, you set the NLUKY parameters to 03323 (N=0, L=3, U=3, K=2, Y=3). When you run Life, there are no resting colors; a cell's color is either off or on. You see a lot of seething activity in Life, but eventually most of the activity dies out. The shapes you see in Life have names given by people who work with cellular automata. For example, you can see *blocks, gliders, blinkers,* and so on.

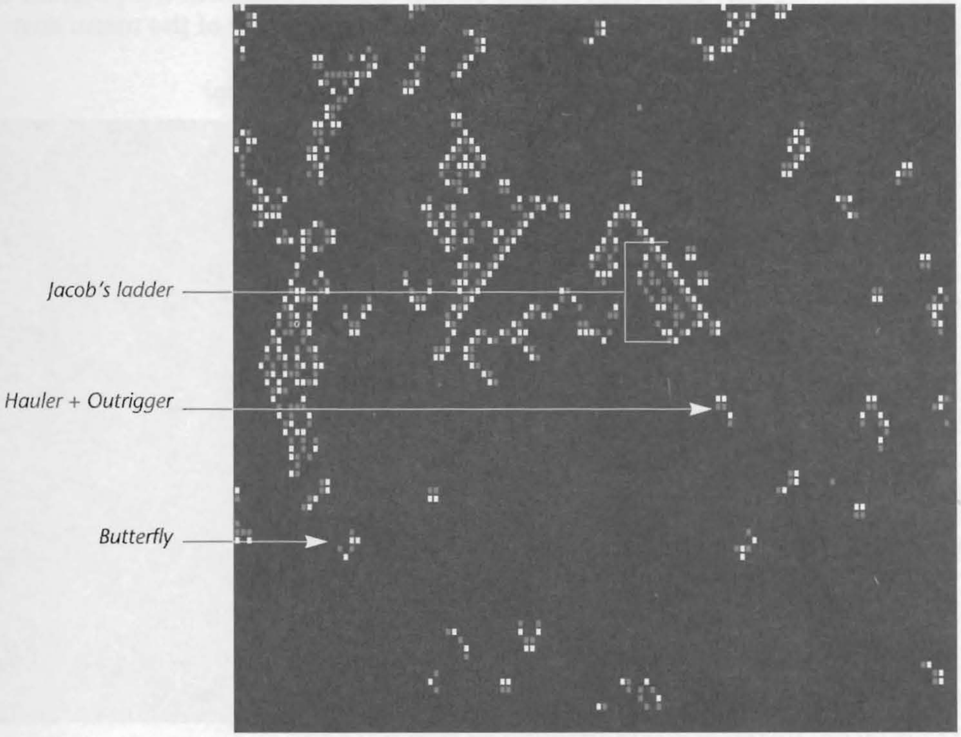
Here's the Life version of the NLUKY rule shown in the image area:



The Brain Version of the NLUKY Rule

In the Brain version of the NLUKY rule, you set the NLUKY parameters to 12299 (N=1, L=2, U=2, K=9, and Y=9). When you run Brain, cells cannot stay on because K and Y are set to 9. (Because a cell has only 8 neighbors, there is no way that it can have 9 neighbor cells on. You cannot count a cell as being a neighbor of itself.) You see a lot of horizontal and vertical movement in Brain. The shapes in Brain, like those in Life, have names. For example, you can see *Jacob's Ladders, haulers, outriggers, butterflies* (the only shapes that moves at 45 degrees), and so on.

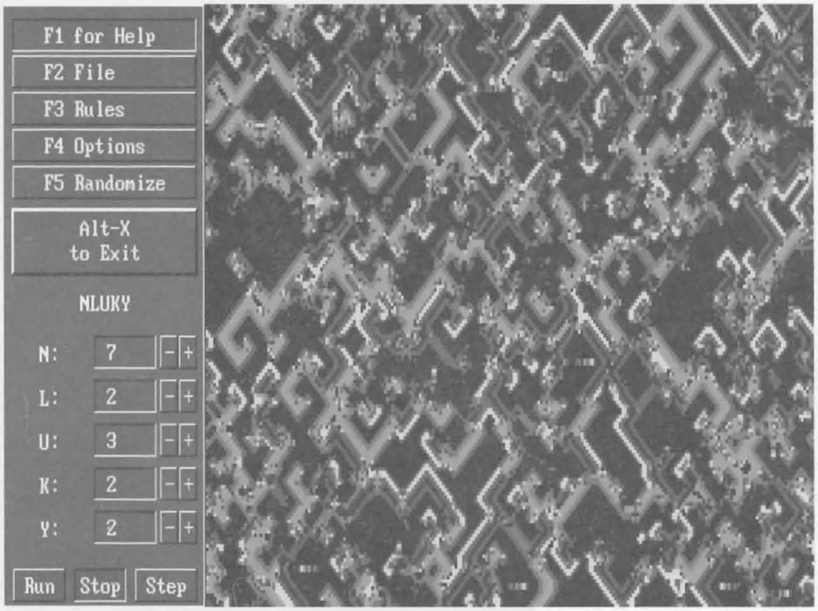
Here's the Brain version of the NLUKY rule shown in the image area:



Changing the NLUKY Parameters

Left-click the F3 Rules button or press F3 until the program displays the NLUKY parameters in the middle of the menu area:

NLUKY Rule image



NLUKY

N: 7 - +

L: 2 - +

U: 3 - +

K: 2 - +

Y: 2 - +

NLUKY parameters

Use the NLUKY parameters to change the NLUKY rule. You can also increase or decrease NLUKY parameters with the + or − buttons next to each entry field.

N

Keyboard shortcut: Shift + N to increase; N to decrease

The N parameter sets the number of resting color states that cells can change to in the NLUKY rule's computation. The default number of resting color states is 7.

- Enter a value between **0** and **127** and press ↵.

L and U

Keyboard shortcut: Shift + L to increase L; L to decrease L; Shift + U to increase U; U to decrease U

The L and U parameters determine when a cell goes from off to on. If a cell is off and stimulus is between L and U, the cell turns on. Otherwise, the cell stays off. The default setting for L is 2; for U, 3.

- Enter a value between **0** and **9** and press ↵.

K and Y

Keyboard shortcut: Shift + K to increase K; K to decrease K; Shift + Y to increase Y; Y to decrease Y

The K and Y parameters determine when a cell changes from the on state to the first of the resting states. If a cell is on and stimulus is between K and Y, the cell remains on. Otherwise, the cell moves into the first resting state. The default setting for both K and Y is 2.

- Enter a value between **0** and **9** and press ↵.

Understanding the Tube Rule

The Tube rule gets its name from the behavior of colonies of tubeworms in coral reefs. Tubeworms live in holes and are easily alarmed. They hide from their neighbors, coming out to eat only when their neighbors are not out.

In the Tube rule, a cell is *hiding,* or in an even-numbered color state, for the number of color-change steps stored in the Hiding parameter. The cell turns on to the first *eating* color state only when it is "safe" to do so. The cell considers it safe if it finds fewer eating neighbors than are stored in the Alarm parameter. If it is safe, the cell stays in the first eating state. Otherwise, the cell begins moving through the successive eating states. The cell changes to each eating color state in succession, up to the number of eating color states stored in the Eating parameter. Once a cell gets to the largest eating color state, it returns to the first hiding state, and the cycle restarts.

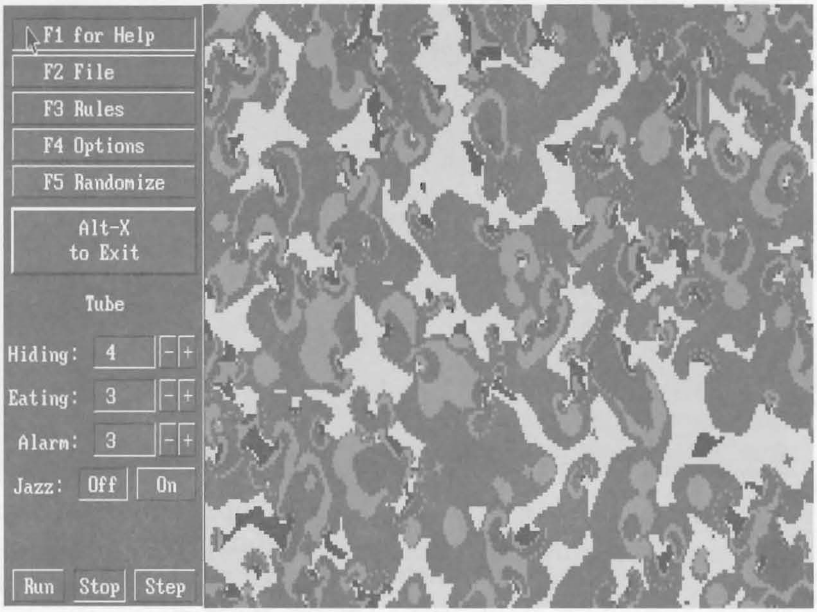
The Jazz condition affects how accurately a cell checks against the Alarm parameter. When you turn this condition on, you see fuzzier edges to the shapes.

For details on the calculations in the Tube rule, see "The Mathematics of the Program" on page 202.

Changing the Tube Parameters and Condition

Left-click the F3 Rules button or press (F3) until the program displays the Tube parameters and condition in the middle of the menu area:

Tube Rule image



Tube parameters

Tube condition

Use the Tube parameters and condition to change the Tube rule. In general, the Tube rule is quite sensitive. Many parameter changes might not look good in this rule. With Jazz turned off, the rule works best with Hiding set to 4, Eating set to 3, and Alarm set to either 2 or 3. With Jazz turned on, the rule works best only with the default settings for Hiding, Eating, and Alarm.

You can increase or decrease the Tube parameters with the (+) or (−) buttons next to each entry field.

Suggestion: You can clarify the action of the Tube rule with the Palette Editor to set all the hiding states, which are all even-numbered colors, to black. To do this, think of the top leftmost color square in the Palette Editor numbered as 1, the one to the right of it numbered as 2, the next one numbered as 3, and so on. Change all even-numbered squares to black and change color square 1 to some other color. Then randomize the Tube rule and watch the hiding states. (You might want to save this palette for later use.)

Hiding

Keyboard shortcut: Shift + H to increase; H to decrease

The Hiding parameter sets the number of color-change steps for a cell to remain in an even-numbered color state. The default is 4.

- Enter a value between **1** and **127** and press ↵.

Eating

Keyboard shortcut: Shift + E to increase; E to decrease

The value in the Eating parameter controls how many color changes a cell makes in the eating state. The default is 3.

- Enter a value between **2** and **127** and press ↵.

Alarm

Keyboard shortcut: Shift + A to increase; A to decrease

When a cell looks to see if it's safe to change to the first eating color state, it checks the number of neighbors that are out, which is stored in the Alarm parameter. If the cell finds that the number of neighbors out is greater than or equal to the number in the Alarm parameter, it considers it unsafe and does not change to the first eating color state. The default alarm number is 3.

- Enter a value between **0** and **9** and press ↵.

Jazz

Keyboard shortcut: J

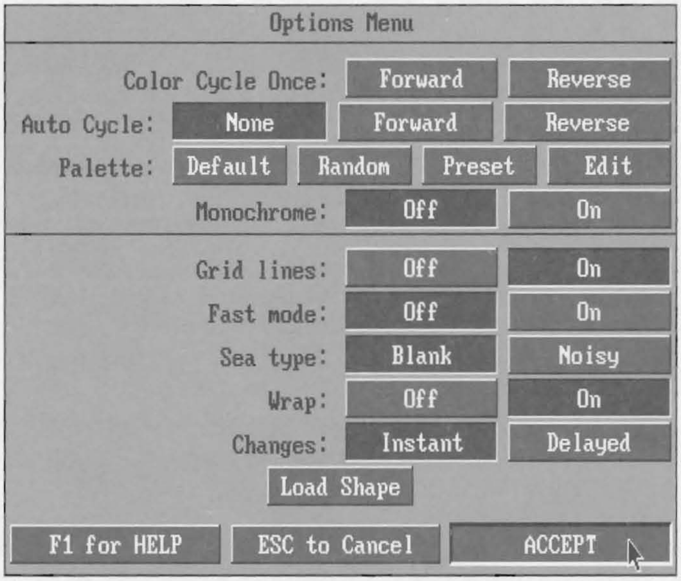
Turning Jazz on "jazzes" the border where alarm sets in, and you see fuzzier shapes. Normally, a cell is alarmed if the number of eating neighbors is greater than or equal to the number stored in Alarm. If Jazz is on, the cell is no longer alarmed by Alarm + 1 neighbors. The rest of the alarm conditions stay the same. The default condition for the Tube rule is Jazz turned off.

- Select the Off or On button to run the Tube rule with an extra alarm component.

Using the Options Menu

With the Options menu, you can turn on color options, grid lines, fast mode, and screen wrap, and you can set a background for very high resolution, set a mode for timing changes to parameters and options, and load various shapes.

To select options, left-click the F4 Options button at the upper-left corner of the screen, or press F4. The Options menu appears:



Color Options

The following color options are common to five of the CHAOS programs. For details on these options, see "Using Color Options" on page 214.

- Color Cycle Once
- Auto Cycle
- Palette (Default, Random, Preset, Edit)
- Monochrome

Grid Lines

Keyboard shortcut: G

Turning on this option lets you see black lines between cells when you zoom in to low and medium resolutions. The default is grid lines turned on.

- Select the Off or On button to turn grid lines off and on for low- and medium-resolution zooms.

Suggestion: Use this option when you want to see color changes more closely.

Fast Mode

Keyboard shortcut: F

This option turns fast mode on and off at low and medium resolutions. With fast mode turned off, the program calculates color changes for both off-screen and on-screen cells.With fast mode turned on, the program calculates color changes only for on-screen cells. So, if you zoom in with fast mode on, and later zoom back out, the center area of the screen is more evolved than the outer areas. The default is fast mode turned off.

- Select the Off or On button to turn fast mode off or on for low- and medium-resolution zooms.

Suggestion: For some rules, zooming in with fast mode on lets you get a BZ-scroll started quickly. The BZ-scroll can then grow outward into the less evolved outer areas when you zoom back out.

Sea Type

Keyboard shortcut: B

Before you zoom out to very high resolution, you can choose a *blank sea* (or black background) that will appear around the small cellular automaton, or you can choose a *noisy* (or randomized) sea. The default sea type is blank.

- Select the Blank or Noisy button to set the background for very-high-resolution zooms.

Suggestion: A blank sea works well with Eat and Tube rules; a noisy sea works well with Hodge and NLUKY rules.

Wrap

Keyboard shortcut: W

This option affects whether cells wrap off screen at the top, bottom, and sides of the cellular automaton. Turning screen wrap on makes cells wrap around at the sides and top and bottom. Turning screen wrap off freezes cells at the current values they have. The default is screen wrap turned on.

- Select the Off or On button to turn screen wrap off and on.

Changes

Keyboard shortcut: F9

This option sets whether the program implements changes you make to parameters and options instantly or after one generation. If you choose instant changes, the program might show half-completed screens when you change to different cellular automata or different resolutions. If you choose delayed changes, the program shows more coherent images but gives slower responses. The default is instant changes.

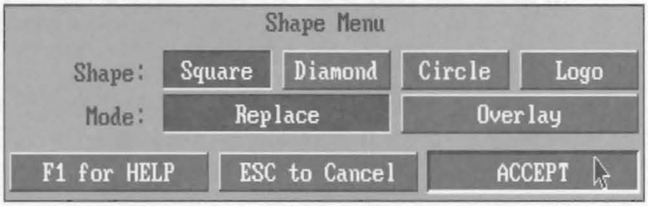
- Select the Instant or Delayed button to set a mode for timing changes to parameters and options.

Suggestion: The distinction between the two settings for this option is most obvious at very high resolution.

Load Shape

Keyboard shortcut: S

When you select the Load Shape option, the Shape menu appears:



Selecting a Shape

This program has four shapes you can load into a rule. To load the shape you want, select one of the following buttons on the Shape menu:

- Square
- Diamond
- Circle
- Logo (loads the Autodesk logo)

Selecting a Shape Mode

When you load a shape, you can select either the replace or overlay mode. In the replace mode, the shape you've chosen is substituted for the full-screen image of the cellular automaton. In the overlay mode, the shape you've chosen is added on top of the full-screen image of the cellular automaton. In both modes, cells within the shape obey the current cellular automaton rule for color changes. Replacing a shape works well for the Hodge and NLUKY rules. Overlaying a shape works well for the Eat and Tube rules. To get the shape mode you want, select one of the following buttons on the Shape menu:

- Replace
- Overlay

The following table summarizes the keyboard shortcuts for overlaying and replacing each shape:

Shape	Overlay	Replace
Square	Alt+1	Alt+5
Diamond	Alt+2	Alt+6
Circle	Alt+3	Alt+7
Logo	Alt+4	Alt+8

Saving and Loading Files

Use the options on the File menu to save and reload parameter files and *.gif* files. For details on saving and reloading files, see "Saving and Loading Files" on page 209.

The Mathematics of the Program

Most cellular automata (CAs) seem to meet one of four fates: they die out, they start repeating themselves, they turn into structureless seething, or they break up into gliders that bounce off each other forever. But there is a fifth option: the BZ-scroll CA reaction.

In the two-dimensional BZ-scroll CAs, the scrolls organize themselves from seeds found at random in the starting pattern. Existing scrolls spin off small child-scrolls that resemble small vortices spawned by large vortices. The boundaries between competing scrolls' terrains are fluid and under constant revision.

In the Toy Universes program, we show four different families of BZ-scroll rules: NLUKY, Hodge, Eat, and Tube. The NLUKY rules are Rudy Rucker's discovery; the Hodge rules were discovered by Martin Gerhardt and Heike Schuster; the Eat rules come from David Griffeath and Bob Fisch; and the Tube rules come from Norman Margolus and Tommaso Toffoli.

All the CAs to be discussed use a two-dimensional grid of squares. The resolution of our grid can be set to very high: 480 by 344, high: 240 by 172, medium: 120 by 86, or low: 60 by 43. Given an arbitrary cell, let C be the positive integer representing the cell's state, and let the new value of the cell that the CA rule calculates be called NewC.

With the exception of the Eat rules, in the rules to be discussed NewC depends on two things: the cell's value C and the value of the sum of the cell's neighbors. Rules of this sort are called "semitotalistic" by CA theorist Stephen Wolfram.

As is customary, we think of the cell as having eight neighbors: the northwest, north, northeast, west, east, southwest, south, and southeast neighbors. We will be interested in two different ways of summing up a cell's neighbors' values to get numbers called the FullEightSum and the FiringEightSum.

The FullEightSum is simply the sum of the eight neighbor states. If the highest usable state is, say, N, then the FullEightSum will be a number between 0 and 8*N. We also define the FullEightAverage as FullEightSum/8.

The idea in forming the FiringEightSum is that a cell's state can be thought of as a binary number, and we can give special significance to certain bit positions. The FiringEightSum focuses on each state's lowest bit. If a neighbor cell's low bit is 1, it contributes 1 to the FiringEightSum, otherwise it contributes 0. The FiringEightSum, in other words, is the sum of the lowest bits of the eight neighbor states. Put differently, the FiringEightSum counts the number of odd-state neighbors that the cell has. The FiringEightSum will always be a number between 0 and 8.

Hodge Rule

The Hodge rules are formulated in terms of two positive integer parameters: N and I. N can be arbitrarily large, and I should be less than N. Typically a value of I approximately equal to N/6 works well. The rule is specified in terms of the FullEightAverage (= FullEightSum/8).

1) The cells have N+1 possible states: 0,1,2,...N.

2) If C = 0 and FullEightAverage = 0 then NewC = 0.
If C = 0 and 0 < FullEightAverage ≤ I then NewC = [I/2] + 1.
If C = 0 and I < FullEightAverage ≤ 2*I then NewC = I.
If C = 0 and 2*I < FullEightAverage then NewC = [I/2] + 1.

3) If 0 < C < N and then NewC = FullEightAverage + I, unless (FullEightAverage + I) > N, in which case NewC = N.

4) If C = N then NewC = 0.

The details of part (2) of the definition are unimportant; many different ways of passing from state 0 to a nonzero state seem to work just as well. [I/2] means "the greatest integer less than or equal to I/2." Note that the Hodge rules are closely related to the simple Rug rule in which NewC = (FullEightAverage + I) MOD N. The crucial difference between Hodge and Rug is that when a Hodge cell value gets larger than N, it is set back to N, and then in the next cycle to 0, whereas in a Rug rule a cell value larger than N is wrapped around to the amount by which N is exceeded. No double-scrolls ever arise in Rug rules.

Eat Rule

The Eat rules are formulated in terms of a single integer parameter N, which our menu calls MaxEat. N can be arbitrarily large. There are two kinds of Eat rules, the deterministic and nondeterministic. The nondeterministic version requires the use of a randomizer to repeatedly select one of a cell's neighbors.

1) The cells have N possible states: 0,1,2,...,N–1.

2) *Deterministic* If any of the cell's neighbors has state (C + 1) MOD N, then NewC = (C + 1) MOD N. Otherwise NewC = C.

or

NonDeterministic If the cell's randomly selected neighbor has state (C + 1) MOD N, then NewC = (C + 1) MOD N. Otherwise NewC = C.

The deterministic Eat rules quickly converge to tight, small spirals that are somewhat monotonous to look at. The nondeterministic rules are more exciting, although the higher the N value, the larger will be the size of the cell grid required for the rule to get started from a uniform random initial condition. With the high resolution grid of 240 by 172, a value of N = 12 is about the largest for which action spontaneously develops. One can, however, force action out of a rule with a large N value in various ways. One trick is to randomize the screen and then set a disk or patch of cells to some single state. You can do this by pressing Alt+1 to overlay a square or Alt+2 to overlay a diamond. Alternately, one can randomize the screen in high resolution mode, then zoom out to very high resolution in the blank sea mode, and see the scrolls grow out of the central spot.

NLUKY Rule

An NLUKY rule is specified by five positive integer parameters: N,L,U,K, and Y. N can be arbitrarily large; the other numbers can range from 0 to 9. (Actually, these numbers could range from 0 to 8; the value 9 is simply used as a signal that a certain possibility is being excluded.) The rule is specified as follows:

1) The cells have N+2 possible states: 0, 1, and the N even numbers 2,4,6,...,2N.

2) Case a) In the case where L ≤ U:
If C = 0 and L ≤ FiringEightSum ≤U then NewC = 1.
If C = 0 and not(L ≤ FiringEightSum ≤U) then NewC = 0.
Case b) In the case where L > U:
If C = 0 and L ≤FiringEightSum OR FiringEightSum ≤U then NewC = 1.
If C = 0 and U < FiringEightSum < L then NewC = 0.

3) Case a) In the case where K ≤Y:
If C = 1 and K ≤ FiringEightSum≤ Y then NewC = 1.
If C = 1 and not(K ≤ FiringEightSum ≤ Y) then NewC = 2.
Case b) In the case where K > Y:
If C = 0 and K ≤ FiringEightSum OR FiringEightSum ≤ Y then NewC = 1.
If C = 0 and Y < FiringEightSum < K then NewC = 0.

4) If 2 ≤C < 2N then NewC = C + 2.

5) If C = 2N then NewC = 0.

The state 0 is called the dead state, state 1, the firing state, and states 2,4,6,...,2N are called the refractory states. The well-known Game of Life rule is the NLUKY rule 03323. Brian Silverman's Brain rule is the NLUKY rule 12299. Rudy Rucker invented the NLUKY categorization to find a common setting for these two rules. (See the *CA Lab* manual for a detailed discussion of these rules.)

Life, on the one hand, seems to gain much of its power from the fact that a firing cell can sometimes keep firing. Brain, on the other hand, gets its richness from the fact that it has a refractory or resting state. Arbitrary NLUKY rules can have both these features.

It was surprising that many of the NLUKY rules are in fact BZ-scroll rules. The simplest example is NLUKY rule 72322,called RainZha. In practice, it seems that almost any rule of the form N2U2Y produces a BZ-scroll pattern, so long as U is greater than 3.

The NLUKY rule N2222 (called Faders) is not a BZ-scroll pattern, but what it does is unique: the pattern grows firing gliders, which come in either a right-handed or a left-handed form. As a right-handed glider moves along, it shoots right-handed gliders out to its right, cumulatively leading to fractal patterns that attempt to become spiral, as the "children" right-handed gliders send out "grandchildren" right-handed gliders, and so on. Spirals fail to develop because a glider's great-grandchildren run into its trail.

Tube Rule

The Tube rules are specified in terms of four positive integer parameters: E, O, A, and Jazz. E is the number of even (hiding) states, O is the number of odd (eating) states, and A specifies a threshold value. We require that O ≥2 and E ≥1. Jazz is either 0 or 1. The rules are formulated in terms of a boolean SafeCondition:

SafeCondition ↔
(FiringEightSum < A) or ((Jazz=1) and FiringEightSum = A+1).

The tube rules can now be specified as follows:

1) The cells have O+E possible states: 1,3,...,(2∗O)–1 and 0,2,4,...,2∗E.

2) If C = 1 and SafeCondition then NewC=1.
If C = 1 and not(SafeCondition) then NewC = 3.

3) If C is odd and 1<C<(2∗O)–1 then NewC = C + 2.

4) If C = (2∗O)–1 then NewC = 0.

5) If C is even and 0 ≤C < 2∗E then NewC = C+2.

6) If C = 2∗E and SafeCondition then NewC = 1.
If C = 2∗E and not(SafeCondition) NewC = 3.

Margolus and Toffoli discuss three variations on this rule. The first two variations take Jazz to be 0 and set EOA to 432 and 433. Their third variation takes EOA to be 433, but changes Jazz to 1. The first of the three reactions converges rapidly to small, square spirals; the second converges more slowly to rounded, octagonal spirals; and the third rule makes very nicely rounded, fuzzy spirals. In general, if E is either O+1 or O+2, and A is 2 or 3 and these rules will produce BZ-scroll patterns.

General Comments

The main thing that all these rules have in common is 1) the existence of a cyclic sequence of states that the cells can move through, and 2) a mechanism that can tend to bring the states of nearby cells into synchronization. The sources of synchronization are as follows:

In the NLUKY rules, a firing cell urges neighboring cells to fire. In the Hodge rule, cell states sit and wait for their fellows for one cycle each time they shoot past the maximum value. In the Eat rules, certain values get copied into their neighbors. And in the Tube rules, a state 1 cell can wait until more than A of its neighbors enter odd states.

Why do the BZ-scroll rules form double scrolls? A double scroll is really a line segment that is radiating a stimulation out of one of the line's sides. The stimulation is more intense at the line's interior points, so the central part of the scroll emanates there. At the line's ends, the stimulation falls off, so the stimulation dies down and gets rounded off in a spiral.

Chapter 8

Saving Files and Using Color Options

This chapter documents features common to all or most CHAOS programs. The features described here are

- Saving and Loading Files
- Using Color Options

Saving and Loading Files

Most CHAOS programs have a File menu that lists options for saving and loading files.

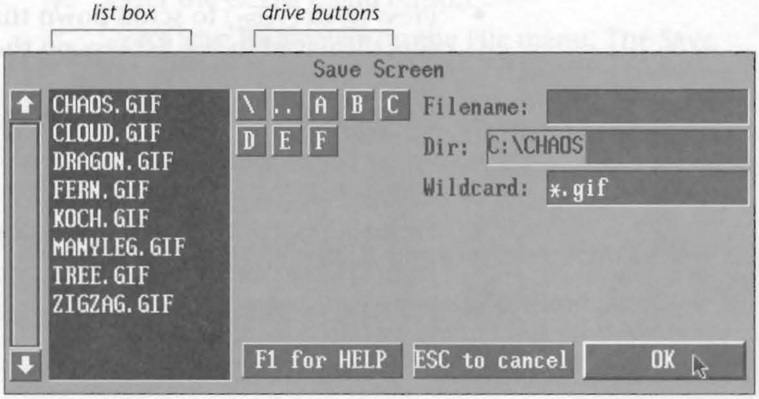
To display the File menu

- Left-click the F2 File button or press (F2).

The File menu in each CHAOS program includes some or all of the File options described in the sections that follow.

Using File Menu Dialogue Boxes

All File menu options use dialogue boxes like this one.



CHAOS.GIF
CLOUD.GIF
DRAGON.GIF
FERN.GIF
KOCH.GIF
MANYLEG.GIF
TREE.GIF
ZIGZAG.GIF

Filename:

Dir: C:\CHAOS

Wildcard: *.gif

Existing files with the extension appropriate for the current file type and program appear in the list box at the left. The Wildcard field shows the default extension used for the list box. For example, if you're saving a GIF file, *.gif appears in the Wildcard field and only files with the extension *.gif* are listed in the box at the left. The Dir field shows the current directory. The Filename field is active and ready to accept your entry.

Working with a Mouse

With a mouse, you can do the following in a File menu dialogue box:

- Scroll through the parameter files displayed in the list box at the left.
- Left-click on an existing filename in the list box to move that name to the Filename field.
- Double-click on a filename in the list box to immediately complete the save or load operation.
- Left-click on a directory button to change to that directory.
- Left-click in the Dir field to enter a new path.
- Left-click F1 for HELP to display the help window for this dialogue box.
- Left-click ESC to Cancel to cancel your entries and exit the dialogue box.
- Left-click OK or press ↵ to confirm your entries and exit the dialogue box.

Working with the Keyboard

Using the keyboard, you can do the following in a File menu dialogue box:

- Press ↓ or PgDn to scroll down through the parameter files displayed in the list box on the left, or press ↑ or PgUp to scroll up.
- Enter a name in the Filename field and press ↵.
- Press Tab to move from the Filename, Dir, Wildcard, and Filename Fields.
- Press F1 to display the help window for this dialogue box.
- Press Esc to cancel your entries and exit the dialogue box.
- Press ↵ to complete the save or load operation and exit the dialogue box.

Saving Parameters

Keyboard shortcut: Alt + S

In each CHAOS program, you can save an image as a set of parameters. The advantage of saving parameters is that the parameter file is small and saves disk space. You can store about one hundred parameter files in the space that one GIF file uses. Also, when you load a parameter file, you can work with that file in the current program. When you load a GIF file, you cannot modify the file or use the program menus or options.

When loading a parameter file, the program recomputes the image from scratch, which can take substantially longer than loading a GIF file.

The parameter files for each CHAOS program have different extensions. For example, Fractal Forgeries parameter files have the extension *.for*, and Chaos Game parameters have the extension *.ifs* (for Iterated Function System). This table shows the programs and their corresponding parameter file extensions:

Program Name	***Parameter File Extension***
The Mandelbrot Sets	*.frp*
Magnets and Pendulum	*.mfg*
Strange Attractors	*.sap*
The Chaos Game	*.ifs*
Fractal Forgeries	*.for*
Toy Universes	*.toy*

To choose the Save Parameters option, follow these steps:

1. Select the F2 File menu button.
2. Select Save Parameters on the File menu. The Save Parameters dialogue box appears. Existing parameter files with the extension appropriate for the current program appear in the list box at the left. The Wildcard field shows the default extension the program uses to save parameters in the current program.

To save the parameters with a new filename, follow these steps:

1. Enter a name for the parameter file and press ↵. (You don't have to add an extension. The program adds the extension displayed in the Wildcard field.)

2. Left-click the OK button or press ↵ to save the file with the new name and exit the dialogue box.

To save parameters to an existing file, follow these steps:

1. Scroll through the parameter files displayed in the list box at the left.
2. Left-click on the existing parameter filename in the list box to move that name to the Filename field, or enter the name in the Filename field.
3. Left-click the OK button or press ↵ to save the parameters to the existing file and exit the dialogue box.

Loading Parameters

Keyboard shortcut: Alt+L

To load a parameter file, follow these steps:

1. Select the F2 File menu button.
2. Select Load Parameters from the File menu. A dialogue box appears listing the available parameter files for the current program.
3. Scroll through the parameter files displayed in the list box at the left.
4. Left-click on the parameter filename in the list box to move that name to the Filename field, or enter the name in the Filename field.
5. Left-click the OK button or press ↵ to exit the dialogue box and load the parameters.

Note: Each CHAOS program has several sample parameter files that you can load to see additional examples of images or layouts. The parameter files for each CHAOS program are saved with an extension unique to each program. For a list of the parameter file extensions, see "Saving Parameters" on page 211.

Saving an Image or the Screen as a GIF File

Keyboard shortcut: Alt+G save image; Alt+H save screen

You can save any CHAOS image as a GIF file. GIF files are large (they take up about 30K each) but they load quickly. GIF files are a common graphic file format compatible with many software programs. You can also bring a GIF file to a graphics service company and have them make a slide or poster out of it.

In many CHAOS programs, you can save a GIF file of the image area, or of the entire screen.

To save the entire screen as a GIF file, follow these steps:

1. Select the F2 File menu button.
2. Select Save screen as GIF from the File menu. A dialogue box appears.
3. Enter a filename and press ↵. (You don't have to enter the *.gif* extension.)
4. Left-click the OK button or press ↵ to save the screen with the new filename.

To save only the image area as a GIF file, follow these steps:

1. Select the F2 File menu button.
2. Select Save image as GIF from the File menu. A dialogue box appears.
3. Enter a filename and press ↵. (You don't have to enter the *.gif* extension.)
4. Left-click the OK button or press ↵ to save the image with the new filename.

Viewing a GIF Image

Keyboard shortcut: Alt + F

To view a GIF file, follow these steps:

1. Select the F2 File menu button.
2. Select View GIF image from the File menu. A dialogue box appears listing the available GIF files.
3. Scroll through the GIF files displayed in the list box at the left.
4. Left-click on a filename in the list box to move that name to the Filename field, or enter the name in the Filename field.
5. Left-click the OK button or press ↵ to exit the dialogue box to display the GIF image on the screen.

When you view a GIF file, you cannot modify the file or use the program menus or options. When you've finished viewing the GIF file, follow these steps to return to the program:

1. Press any key to return to the View GIF image dialogue box.
2. Press Esc to return to the program.

Using Color Options

The following color options are common to most CHAOS programs and appear on the Options menu.

Color Cycle Once

Keyboard shortcut: Alt+J cycle forward; Alt+K cycle reverse

The Color Cycle Once option lets you change the colors in the image one step at a time in a forward or reverse direction.

To change colors one step at a time, select one of the following buttons next to the Color Cycle Once option:

- Forward
- Reverse

Auto Cycle

Keyboard shortcut: Alt+Y change to forward, reverse, none

The Auto Cycle option spins the colors continuously in a forward or reverse direction. The default setting is color cycle turned off.

To cycle continuously through colors, select one of the following buttons next to the Auto Cycle option:

- Forward
- Reverse

To turn off Auto Cycle, do one of the following:

- Select the None button next to the Auto Cycle option.
- Press Alt+Y until Auto Cycle is off.

Palette

The Palette option gives you several ways to use different sets of colors for CHAOS images.

To change the palette, select one of the following buttons next to the Palette option:

- Default
- Random
- Preset
- Edit

Each palette option is described in the following sections.

Default Palette

Keyboard shortcut: Alt + D

The Default palette option resets the default display colors or the palette saved in the *.pal* file for the current program. (For more information about the *.pal* file, see "Changing the Default Palette" on page 218.)

Random Palette

Keyboard shortcut: Alt + B

The Random Palette option sets random display colors.

Preset Palettes

Keyboard shortcut: Alt + A

The Preset Palettes option lets you cycle through the preset palettes of display colors for the current program.

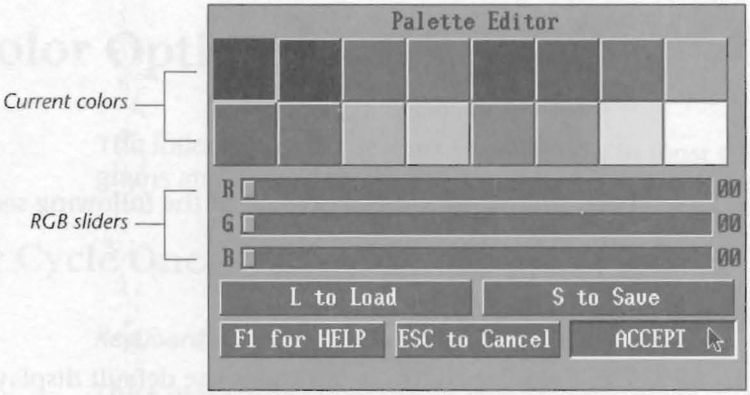
Edit Palette

Keyboard shortcut: Alt + P

In addition to letting you select random or preset palettes, CHAOS lets you create your own palette of 16 colors. CHAOS uses the palette for the image and other screen elements. Use the Palette Editor to edit palettes and to save and reload them.

To use the palette editor, follow these steps:

1. Select the Options (or Opts) menu button to display the Options menu.
2. Select the Edit button next to the Palette option. The Palette Editor appears:



Current colors

RGB sliders

To change the colors in the palette, follow these steps:

1. Move the highlight box to the color you want to change by doing one of the following:
   - Left-click on the color.
   - Press ↓ or ↑ to move to the row of colors. Press → or ← to move to the color you want to change. Then press ↵.
2. Move the RGB Slider to the value you want by doing one of the following:
   - Point to a new location on a slider bar and left-click.
   - Press ↓ to move down to the next slider bar or press ↑ to move to the previous slider bar. At the slider bar you want to change, press → or ← to adjust the slider bar to the color-mix you want.

   The highlighted color changes as you adjust each slider bar. You can also see the effect on the program screen.
3. Repeat steps 1 and 2 for each color you want to change in the palette.
4. Select the ACCEPT button to accept the changes and return to the Options menu.

Note: To cancel all changes, select the ESC to Cancel button, or right-click, or press Esc.

5. Select the ESC to Cancel button, or right-click, or press Esc to exit the Options menu.

Saving and Loading a Color Palette

When you change a color palette, the new palette exists in memory and won't be available after you quit the program unless you save the palette. You can save any number of palettes to unique files that have the extension *.pal*. Then you can load the palette at any time.

Note: For additional information on using save and load dialogue boxes, see "Using File Menu Dialogue Boxes" on page 209.

To save a palette, follow these steps:

1. Select the Edit button next to the Palette option on the Options menu. The Palette Editor appears.
2. Press S or left-click the S to Save button. The Save Palette dialogue box appears.
3. Enter the name of the palette file you want to save and press ↵. (You don't have to enter the *.pal* extension.)
4. Left-click OK or press ↵ to save the palette with the new name.

To load a palette, follow these steps:

1. Select the Edit button next to the Palette option on the Options menu. The Palette Editor appears.
2. Press L or left-click the L to Load button. The Load Palette dialogue box appears listing the available palette files.
3. Scroll through the palette files displayed in the list box at the left.
4. Left-click on a palette filename in the list box to move that name to the Filename field, or enter the name in the Filename field.
5. Left-click the OK button or press ↵ to exit the dialogue box and load the palette.

Changing the Default Palette

Each CHAOS program (except Fractal Forgeries) has at least one default palette. Toy Universes has four default palettes. You can change the default palette in any program by editing a palette and saving it to the default palette filename. The default palette filenames for the CHAOS programs are:

Program Name	***Default Palette Filename(s)***
Mandelbrot Sets	*mandel.pal*
Magnets and Pendulum	*magnets.pal*
Strange Attractors	*attract.pal*
The Chaos Game	*game.pal*
Toy Universes	*hodge.pal*
	eat.pal
	nluky.pal
	tube.pal

Suggestion: Before you change a default palette, you might want to save the original default palette file to another name such as mandel1.pal. That way, you can always rename that file to the original default palette filename for the program.

To change the default palette, follow these steps:

1. Select the Edit button next to the Palette option on the Options menu. The Palette Editor appears.
2. Change the palette as described in "Edit Palette" on page 215.
3. Press S or left-click the S to Save button. The Save Palette dialogue box appears.
4. Enter the default palette name of the CHAOS program you're using.
5. Left-click OK or press ↵ to save the palette as the default palette.

Monochrome

Keyboard shortcut: Alt+M

If you have a VGA monitor, the Monochrome option lets you change the screen colors to black and white (or another two-color combination if you've edited your color palette) and then back to multiple colors. The default setting is Monochrome turned off.

- To toggle between monochrome and multiple colors, select the On or Off button next to the Monochrome option.

Suggestion: You might want to use a monochrome black and white setting when you save a *.gif* image that you want to print. The printed black and white image might be crisper than a color one.

Extending the Color Palette

Keyboard shortcut: Alt+E to toggle on and off

The default setting for all CHAOS programs (except Toy Universes) reserves five colors for the program menus and buttons. These reserved colors ensure that menus and buttons remain readable when you randomize the palette or use color cycling. However, these reserved colors are not available for CHAOS images.

You can "unlock" the reserved menu colors to extend the palette of colors available for CHAOS images. If you unlock these colors, you might find that menus and buttons are difficult to read in some palettes. You might want to wait until you're familiar with a program (and its keyboard shortcuts) before you experiment with extending the color palette.

- Press Alt+E to extend the palette.

After you extend the palette, images can use all available colors except black, which is reserved for the background. You'll notice a change when you use color cycling options or when you randomize the palette. Extending the palette in The Mandelbrot Sets program produces the most dramatic change in the images.

- Press Alt+E again to return to reserved menu colors.

Suggestion: If you're not sure if palette extending is on or off, press Alt+B to randomize the palette. If the menu buttons change colors, then palette extending is on. To quickly return to the default palette (with readable menus), press Alt+D.

The Genesis of James Gleick's CHAOS: The Software

I met James Gleick in September, 1987, at the first-ever workshop on Artificial Life, held in Los Alamos. He was covering the conference for the *New York Times*, and I was there as both a San Jose State University mathematics professor interested in cellular automata and the author of two cyberpunk science-fiction novels about intelligent robots.

Chaos: Making A New Science was published the next month. I happened to be assigned the job of reviewing it for the Washington Post Book World. I enjoyed the book very much; I found that it made a lot of frighteningly abstruse concepts understandable. Like many people, I'd been wanting to start thinking about chaos, and this popularly written book gave us the courage to try. Presciently enough, my review said, "Chaos could well prove to be something of a scientific best-seller."

A year later I was working for Autodesk, Inc., helping develop a software package that would be called CA Lab: Rudy Rucker's Cellular Automata Laboratory. James Gleick phoned Autodesk with a proposal for a software package illustrating some of the chaotic processes described in his book. To my delight, Autodesk agreed. If all goes well, CA Lab and CHAOS: The Software may be Autodesk's first two products in an ongoing Science Series.

James Gleick and I met near Disneyland in the spring of 1989 to discuss the kinds of programs we would put in, and then I spent most of the next year writing C and Assembly Language code for five of the programs: The Mandelbrot Sets, Magnets and Pendulum, Toy Universes, The Chaos Game, and Strange Attractors. John Walker supplied the code for the Fractal Forgeries program. Starting late in 1989, Joshua Gordon began putting interfaces onto the programs and, where necessary, reworking the code, sometimes quite extensively. Throughout 1990, Chaos the Software continued to evolve, with Josh and I sparing no effort to make everything excellent, often guided by James Gleick's many valuable suggestions.

The present manual is a hybrid work. James Gleick wrote the beginning of the chapters, and I wrote the mathematics sections at the chapter ends, with Autodesk technical writers creating the sections that lie between. Many other people at Autodesk helped on this project as well.

Being involved in such a complex software development project has been an amazing experience for me. One after another of our chaotic simulations has come to life and become fully interactive, and each one has in turn taken over my consciousness for weeks at a time. One month everything looked like a cubic Julia set, the next month everything was a Barnsley fractal, after that music began sounding like Hénon attractors, then mountains became Fourier synthesized, and this month political trends look like the oscillations of a pendulum in a field of random magnets. After thousands of hours with our CHAOS programs, I still keep discovering new things.

I'm happy to have helped make these wonderful mathematical forms easier to play with. Have fun!

Rudy Rucker
November 14, 1990

Introducing Rudy Rucker

Rudy Rucker was born March 22, 1946 in Louisville, Kentucky. He graduated from Swarthmore College and received a Ph.D. in Mathematics from Rutgers University. Rucker is best known for his popular books on mathematics and for his science fiction novels, including *Software* and *Wetware*, each of which won the Philip K. Dick Award. Rucker is Mathenaut in the Advanced Technology department of Autodesk Inc. and a professor of Mathematics and Computer Science at San Jose State University. Rucker recently published his thirteenth book, *The Hollow Earth*, a science-fiction adventure novel that starts in 1836 and features Edgar Allan Poe. Rucker lives in Los Gatos, California, with his wife Sylvia and their three college-age children.

Bibliography

Abbott, Edwin. 1884. *Flatland: A Romance of Many Dimensions.*

Abraham, Ralph. 1982–1988. *Dynamics* (Vol. 1–4), Santa Cruz: Aerial Press.

Barnsley, Michael F. 1988. *Fractals Everywhere.* San Diego: Academic Press.

Branner, Bodil, and Hubbard, John. 1988. The iteration of cubic polynomials. Part I: The global topology of parameter space. *Acta Mathematica* 160, pp. 143–206.

Carpenter, Loren. 1984. Cited in Mandelbrot, Benoit, et. al. Additional Perspectives on Fractals. *The College Mathematics Journal,* March, pp. 115-119.

Crutchfield, James; Farmer, Doyne; Packard, Norman; and Shaw, Robert. 1986. Chaos. *Scientific American,* December, pp. 46–57.

Devaney, Robert. 1989. *An Introduction to Chaotic Dynamical Systems.* Menlo Park: Addison-Wesley.

Dewdney, A. K. 1987. Computer Recreations: Probing the Strange Attractors of Chaos. *Scientific American,* July, pp. 108–111.

Dewdney, A. K. 1988. *The Armchair Universe.* New York: W.H.Freeman & Company.

Dewdney, A. K. 1988. Computer Recreations: The hodgepodge machine makes waves. *Scientific American,* August, pp. 104–107.

Dewdney, A. K. 1988. Beauty and Profundity: The Mandelbrot Set and a Flock of Its Cousins Called Julia. *Scientific American,* November, p. 140.

Dewdney, A. K. 1989. Computer Recreations: A cellular universe of debris, droplets, defects and demons. *Scientific American,* August, pp. 102–105.

Dewdney, A. K. 1990. *The Magic Machine: A Handbook of Computer Sorcery.* New York: W.H.Freeman & Company.

Dewdney, A. K. 1990. Mathematical Recreations: How to transform flights of fancy into fractal flora and fauna. *Scientific American*, May, pp. 126–129.

Douady, Adrian, and Hubbard, John. 1985. On the Dynamics of Polynomial-like Mappings. *Annales Scientifique de l'Ecole Normale Supérieur* 18:287.

Feigenbaum, Mitchell J. 1978. Quantitative Universality of a Class of Nonlinear Transformations. *Journal of Statistical Physics* 19:25.

Feigenbaum, Mitchell J. 1980. Universal Behavior in Nonlinear Systems. *Los Alamos Science*, Summer, p. 4.

Gerhardt, Martin; Schuster, Heike; and Tyson, John. 1990. A Cellular Automaton Model of Excitable Media Including Curvature and Dispersion. *Science* 247, March 30, pp. 1563–1566.

Gleick, James. 1987. *Chaos: Making a New Science*. New York: Viking.

Grebogi, Celso; Ott, Edward; and Yorke, James. 1985. Attractors on an N-Torus: Quasiperiodicity Versus Chaos. *Physica* 15D, pp. 354–373.

Grebogi, Celso; Ott, Edward; and Yorke, James. 1987. Chaos, Strange Attractors, and Fractal Basin Boundaries in Nonlinear Dynamics. *Science,* Vol. 238, October 30, pp. 632–638.

Griffeath, David. 1988. Cyclic Random Competition. *Notices of the American Mathematical Society* 35, pp. 1472–1480.

Hénon, Michel, and Heiles, C. 1964. The Applicability of the Third Integral of the Motion: Some Numerical Experiments. *Astronomical Journal* 69:73.

Hénon, Michel. 1976. A Two-Dimensional Mapping with a Strange Attractor. *Communications in Mathematical Physics*,50:69.

Hofstadter, Douglas. 1985. *Metamagical Themas: Questing for the Essence of Mind and Pattern*. New York: Basic Books.

Jurgens, Hartmut; Pietgen, Heinz-Otto; and Saupe, Dietmar. 1990. The Language of Fractals. *Scientific American*, August, pp. 60–67.

Li, T.-Y., and Yorke, James. Period Three Implies Chaos. *American Mathematical Monthly* 82:985.

Lorenz, Edward N. 1963. Deterministic Nonperiodic Flow. *Journal of the Atmospheric Sciences* 20:130.

Mandelbrot, Benoit. 1982. *The Fractal Geometry of Nature*. San Francisco: W. H. Freeman.

Margolus, Norman, and Toffolli, Tommaso. 1987. *Cellular Automaton Machines*. Cambridge: MIT Press.

May, Robert. 1976. Simple Mathematical Models with Very Complicated Dynamics. *Nature* 261:459.

Moon, Francis. 1987. *Chaotic Vibrations*. New York: John Wiley & Sons.

Peitgen, Heinz-Otto, and Richter, Peter. 1986. *The Beauty of Fractals: Images of Complex Dynamical Systems*. New York: Springer-Verlag.

Peitgen, Heinz-Otto, and Saupe, Dietmar. 1988. *The Science of Fractal Images*. New York: Springer-Verlag.

Pickover, Clifford. 1980. *Computers, Patterns, Chaos and Beauty: Graphics from an Unseen World*. New York: St. Martin's Press.

Press, William; Flannery, Brian; Teukilsky, Saul; and Vettering, William. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge: Cambridge University Press.

Rucker, Rudy. 1984. *The Fourth Dimension: Toward a Geometry of Higher Reality*. Boston: Houghton Mifflin.

Rucker, Rudy. 1987. *Mind Tools: The Five Levels of Mathematical Reality*. Boston: Houghton Mifflin.

Rucker, Rudy. 1987. Patterns of Disorder (Book review of James Gleick's *Chaos: Making a New Science*). *Washington Post Book World*. November 1, p.3 and p.8.

Rucker, Rudy, and Walker, John. 1989. *CA Lab: Rudy Rucker's Cellular Automaton Laboratory*. Sausalito, CA: Autodesk, Inc.

Ruelle, David, and Takens, Floris. 1971. On the Nature of Turbulence. *Communications in Mathematical Physics* 20:167.

Thompson, J. M. T., and Stewart, H. B. 1986. *Nonlinear Dynamics and Chaos*. New York: John Wiley and Sons.

Voss, Richard F. 1985. Random Fractal Forgeries, in Earnshaw et. al. *Fundamental Algorithms for Computer Graphics*. Berlin: Springer-Verlag.

Winfree, Arthur. 1974. Rotating Chemical Reactions. *Scientific American*, June, pp. 82–95.

Wolfram, Stephen. 1986. *Theory and Applications of Cellular Automata*. Singapore: World Scientific.

Index

Acknowledgements

Autodesk gratefully acknowledges the following pioneering scientists and mathematicians and the many others whose discoveries made this software possible:

Ralph Abraham
Michael Barnsley
Mitchell Feigenbaum
Michel Hénon
John Hubbard
Edward Lorenz
Benoit Mandelbrot
Robert May
Heinz-Otto Peitgen
Clifford Pickover
Andrzej Proskurowski
Peter Richter
David Ruelle
Dietmar Saupe
Stephen Smale
Floris Takens
Stephen Wolfram
James Yorke

Package and manual cover art, *Julia Set Bounding Four Basins on Riemann Sphere* from *The Beauty of Fractals*, H.O. Peitgen and P.H. Richter, 1986, Springer-Verlag, New York.