SESSION 7 PERCEPTION

Moderator—William S. Maki, North Dakota State University

Simple geometric fractals

DARRELL L. BUTLER
Ball State University, Muncie, Indiana

Recently, vision scientists have begun to explore fractals. This paper describes a set of programs that can be used to create fractal and fractal-like drawings. The programs were implemented on the Apple II series of computers. The programs were primarily designed to create deterministic and random fractal-like patterns with fractal dimensionality between 1 and 2. A supplementary program computes the box dimensionality, a measure of dimensionality that does not assume an infinite recursive process. The advantages of this measure of dimensionality over the more typical self-similar measure are discussed.

The field of fractal geometry is new and is growing at a rapid rate. It concerns shapes that are (or could have been) created by a sequence of operations, often iterations. Mathematicians and computer scientists have proposed that fractal geometry provides a valuable model for complex natural shapes (Mandelbrot, 1982; Peitgen & Richter, 1986; Peitgen & Saupe, 1988) because most natural shapes are the result of a sequence of operations. Perhaps because fractal geometry provides methods for quantitatively describing natural shapes, vision scientists in psychology have begun to explore them. Cutting and Garvin (1987) investigated the relation between various fractal variables and judgments of complexity. Marchak (1987, 1989, 1990) explored fractal models in regard to perception of textures and surfaces in nature. Gilden and Schmuckler (1989) studied discrimination of fractal contours. Results of these studies suggest that fractal geometry may have some value, but much more work is needed before an evaluation of the relevance of fractal geometry to human perception can be made.

To study the perception of fractals, scientists need tools that can assist in the creation of fractal patterns. Computers with appropriate software provide such a tool.

The author would like to thank Doug Brown, Kathy Kirkhoff, and Teresa Dinsmore for assisting with the development and debugging of early versions of the software and Danica Radivojevic for aid in debugging the most recent version. For reprints, contact the author at the Department of Psychological Sciences, Ball State University, Muncie, IN 47306, or 00dlbutler@BSUvax1.bitnet. For copies of the program, please send a 5.25-in. diskette and diskette mailer.

Some software designed specifically to generate fractals has become available, for example, FractalMagic (Bolme, 1987). One problem with most programs currently available is that they are primarily of value in exploring some specific mathematical fractal sets, such as the Mandelbrot and Julia sets. Although these fractal sets are of particular value to mathematicians, they are not as useful to perception researchers because they do not facilitate the development of experimental stimuli with well-defined characteristics along various dimensions.

This paper describes a set of programs that can be used to create fractal and fractal-like stimuli and that create drawings such as those shown in Figures 1 and 2. The programs were first implemented on the Apple II. Conversion and enhancement of the programs to run on IBM PCs (in the C language) is in progress.

In these programs, fractals and fractal-like patterns are iteratively constructed using two shapes, an initiator (or starting shape) and a generator (or replacement pattern). The initiator is a line, the length and orientation of which are controlled by the user. One way to think about the creation of a fractal is to replace the initiator by a pattern of lines (the generator), then replace each resulting line by a scaled-down version of the pattern, continuing this process iteratively. In this paper, the number of replacements is referred to as the number of nestings in a picture or drawing. Figure 1 illustrates the concepts of initiator, replacement pattern, and nestings using a well-known fractal, the Koch snowflake.

Technically, fractals are created using an infinite number of nestings, or at least a large number. The term

Figure 1. Illustration of the iterative process that creates the Koch snowflake and some of the key terms used to describe the process, that is, initiator, replacement pattern, and nestings.

Figure 2. Three simple patterns that can be easily created using the programs described here.

“fractal-like” is used to refer to drawings that are created using a small number of nestings. Fractal-like drawings may be particularly useful in perception research because they provide a link between natural shapes, which may be well-modeled by fractals, and the simple line patterns that are more typical of stimuli used in psychological research.

The program described here can produce both deterministic and random fractal-like images. Deterministic fractals constructed from a particular initiator and generator are always the same. The patterns shown in Figures 1 and 2 are deterministic fractals. The creation of random fractals involves a random process. Thus, a set of drawings created from a particular initiator and generator appear slightly different. Figures 3 and 4, which are described in greater detail below, show some random fractals that are related to the Koch snowflake shown in Figure 1.

The package consists of a menu program (INITIALIZE) that is loaded when the system is booted, a main fractal

program (FRACTALS), and several auxiliary programs (BOX DIMENSION and printer drivers). The programs can be run by being selected from an opening menu or run from the operating system. Each of the programs is described below.

FRACTALS (VERSION 4.1)

The fractals program is menu driven. In the opening menu, there are six options: create/edit, draw, disk, print picture, list pattern information, and exit. Each of these options is described below.

Create/Edit

This submenu provides three options: change initial line, create/edit replacement pattern using number editor, and create/edit replacement pattern using graphics editor.

Initial line. The initial line can be set to any length and any orientation that can be drawn on a high-resolution

Figure 3. Some fractal-like patterns produced using randomness in the line-drawing algorithm. The patterns have maximum randomness set at 50% of line length (top pattern) and 100% of line length (bottom pattern).

graphics page (192×278). The user enters the x - and y -coordinates of the ends of the line. To simplify programming of the drawing, the second end point of the line cannot be to the left of the first end point.

Number editor. Replacement patterns contain between 2 and 20 lines. In the number editor, they are defined by angle changes (-180° to $+180^\circ$, where $+$ is clockwise). For example, the angle changes needed to create the replacement pattern for the Koch snowflake (Figure 1) are 0° , 60° , -120° , and 60° . As the program draws this pattern, these angles are interpreted to mean: Draw the first line in the same orientation as the initiator, then turn 60° counterclockwise and draw the next line, then turn 120° clockwise and draw the next line, and finally turn 60° counterclockwise and draw the last line. Length of replacement lines is not entered because it is computed by the program to make the replacement pattern fit the initial line and each subsequent line to be replaced. All 20 possible angles are shown on the screen. Arrow keys are used to move among them. The D key deletes an angle and the I key inserts an angle between previously defined angles. The E key erases all defined angles. Escape (ESC) is pressed at the end of editing for a return to the main menu. If fewer than two angles have been defined, the program automatically resets the parameters to the Koch snowflake.

Graphics editor. The graphics editor provides an alternative method of creating replacement patterns. A triangle that points in the last direction drawn (or 3 o'clock if no lines have been drawn) is shown on the graphics screen. Six commands are printed at the bottom of the screen, and the arrow keys and the return key are used to select from among them. The first command is TURN AND DRAW. If this option is selected, the user enters the number of degrees to turn. The second command is SET ANGLE AND DRAW. This command differs from the first in that the angles are defined with reference to the screen

(3 o'clock is 0°) rather than with respect to the last line drawn. The third command is DELETE LINE; it deletes the last line drawn. The fourth command is ERASE ALL, which erases all lines. The fifth command is CHANGE SETTINGS. The user can reset the starting position of drawing and the lengths of lines drawn. The final command is FINISHED. When this command is chosen, the program returns to the main menu.

Draw

To create a fractal-like drawing from the initiator and replacement patterns, the user must enter three values that affect the drawing algorithm. First, the user decides on number of nestings (0-6). The other two values concern the types and degree of randomness. The user must indicate the maximum amount of randomness as a percentage of the line length in the drawing. Zero is selected for a deterministic fractal, and a positive value is chosen for a random fractal-like drawing. Figure 3 shows the Koch snowflake when drawn with 50% and 100% maximum randomness in line lengths. Then the user must indicate the maximum size of a random angle to be selected between each drawn line. Figure 4 shows the Koch snowflake with 30° and 60° maximum random angles.

A drawing is then created on a high-resolution graphics screen. (The Koch snowflake is drawn if no other pattern has been defined.) If the drawing does not fit on the graphics screen, the drawing will not be completed and two relatively high-pitch tones are produced. If the drawing is completed without error, a relatively low-pitch clicking sound is produced. After either one of the sounds, the user presses any key to return to the main menu.

Disk

This submenu lets users obtain catalogs of disks, save or retrieve a picture of drawing (a 34-sector binary file), and save or retrieve the description of a pattern (a 2-sector ASCII file). The binary files, which can be used in experiments, printer dump programs, and so forth, are use-

Figure 4. Some fractal-like patterns produced using randomness in the angle-change algorithm. The patterns have maximum randomness set at 30° (top pattern) and 60° (bottom pattern).

ful for saving drawings of fractals that take a long time to create. Such a file is needed to calculate the box dimensionality (described below). When a binary file is saved, an ASCII file of the same name (with an .INFO extension) is also created. Later, when the binary file is reloaded, so is the ASCII file. The ASCII file contains the number of angles in the replacement pattern, their values, number of nestings, line length, fractal dimensionality, and the four coordinates defining the initial line. The ASCII files have only one number per line and could easily be generated by using a text editor.

Print

The most recently created picture is sent to the printer using the Beagle Bros. printer dump routine.¹ The picture may be printed in full- or half-page width, and the user can indicate whether pattern information should also be printed. If it is, then the number of nestings is printed, followed by all information available in the list menu option described below.

List

The following information is listed on the screen (and printer if desired): self-similarity, number of nestings (default = 1), number of lines in pattern, length of lines in pattern, beginning and ending x - y -coordinates of initial line, and the angles defining the replacement pattern. Self-similarity is a common fractal statistic. It is well defined and easy to calculate if the lines in the replacement pattern are all of the same length. Let N = the number of lines in the replacement pattern and r = the length of lines in the replacement pattern relative to the length of line to be replaced. Then fractal dimensionality (d) is

$$d = \log(N)/\log(1/r).$$

Exit

If this option is selected from the fractals menu, the program ends and the user is returned to the opening, or boot, program.

BOX DIMENSIONALITY

In the fractal program described above, the only computed statistic about the pattern is self-similarity. Self-similarity is often interpreted as fractal dimensionality. In fractal geometry, a single straight line has a dimensionality of 1, a two-dimensional surface has a dimensionality of 2, and a homogeneous solid has a dimensionality of 3. But shapes can have other dimensionalities. For example, the Koch snowflake is, in some sense, more than a single straight line but less than a two-dimensional surface. In fact, its dimensionality is about 1.262.

Self-similarity is not a particularly appropriate measure of dimensionality for many of the patterns that the program can produce. One of the most severe problems concerns fractal-like stimuli. The dimension of self-similarity is the fractal dimension if *the number of nestings is in-*

finite. For fractal-like stimuli, especially those created using just one or two nestings, this may not be a reasonable measure of fractal dimension. There is also a major problem for drawings created using randomness in the process. The self-similarity dimension is only an approximation of the dimensionality of patterns created using a process with randomness. If the pattern has substantial randomness, self-similarity may be a poor approximation. A third major problem concerns self-intersecting patterns, both random and deterministic. Some patterns do not self-intersect—for example, the Koch snowflake. But it is easy to create patterns that do. Such patterns “overlap” or draw on the same parts of the screen more than once. Self-similarity does not provide an accurate measure of the dimensionality of patterns that have self-intersections.

There are measures of dimensionality that do not have the problems associated with self-similarity. One broadly applicable measure is the box dimension. The box dimension (d) is defined by the equation:

$$N = cB^{-d},$$

where N is the number of “boxes” having size B that are required to cover the drawing.

The box dimension cannot be calculated as exactly as can the dimension of self-similarity, but it can be approximated. The program BOX DIMENSION does this by dividing the graphic screen into rectangles of various sizes (ranging from the size of a single pixel up to boxes containing 3,584 pixels). The program counts the number of boxes of each size that include a part of the drawing. The box dimension is then approximated by finding a best fitting, least-squares solution to the logarithmic transforms of the box equation. The program approximates both the box dimension (d) and the constant (c) and provides a 95% confidence interval for the box dimension.

The box dimensions were calculated for the Koch snowflakes with different numbers of nestings shown in Figure 1. The dimensions were 1.0001, 1.041, 1.085, 1.170, and 1.233 for the five patterns. Note that the self-similarity measure is 1.262 for all of these patterns except the first one.

PROGRAM INFORMATION

The programs were written in Applesoft Basic in Dos 3.3. The programs have been tested on a variety of Apple II computers with at least 48K RAM and on one Apple compatible, a Franklin 1200. A separate file (REM.FRACTALS) contains the definition of variables and functions of program sections. Users should be able to alter these programs to fit specific needs, replace parts with machine code, or transfer to ProDos. For example, those who do not own Beagle Bros. printer routines should be able to substitute their own print routines. Also, it would not be difficult to create initiators more complex than a single line.

Researchers who have LOGO may find little advantage in using these programs. Among the advantages of these programs is that they are very user friendly, especially with

regard to creating a drawing, they calculate box dimensionality and self-similarity, and they provide some other kinds of information that may be useful to researchers. However, they are generally slower than LOGO in producing most drawings.

REFERENCES

- BOLME, M. W. (1987). *FractalMagic* [Computer program]. Bellevue, WA: Sintar Software. (P.O. Box 3746, zip code 98009, phone 206-455-4130)
- CUTTING, J. E., & GARVIN, J. J. (1987). Fractal curves and complexity. *Perception & Psychophysics*, **42**, 365-370.
- GILDEN, D., & SCHMUCKLER, M. (1989, November). *The perception of fractal contour*. Paper presented at the annual meeting of the Psychonomic Society, Atlanta, GA.
- MANDELBROT, B. B. (1982). *The fractal geometry of nature*. San Francisco: Freeman.
- MARCHAK, F. M. (1987). *Fractal models in the visual perception of textures and surfaces in nature*. Unpublished doctoral dissertation, Dartmouth College, Hanover, New Hampshire.
- MARCHAK, F. M. (1989, November). *Fractal geometry as a model of unpatterned texture perception*. Paper presented at the annual meeting of the Psychonomic Society, Atlanta, GA.
- MARCHAK, F. M. (1990, November). *Perceptual discrimination of fractal textures*. Paper presented at the annual meeting of the Psychonomic Society, New Orleans, LA.
- PEITGEN, H.-O., & RICHTER, P. H. (1986). *The beauty of fractals*. New York: Springer-Verlag.
- PEITGEN, H.-O., & SAUPE, D. (1988). *The science of fractal images*. New York: Springer-Verlag.

NOTE

1. Beagle Bros. software is copyrighted. Users may contact Beagle Bros. or substitute other printer drivers.