Modelling of 3D fractal images

Bulusu Rama[1] and Jibitesh Mishra[2]

[1] SLC's Institute of Engg. & Technology
Hyderabad, India

[2] College of Engineering & Technology, BPUT
Bhubaneswar, India

ABSTRACT— ***Fractals provide an innovative method for generating three-dimensional (3D) images of real-world objects by using computational modelling algorithms based on the imperatives of self-similarity, scale invariance, and dimensionality. Images such as coastlines, terrains, cloud mountains, and most interestingly, random shapes composed of curves, sets of curves, etc. present a multi-varied spectrum of fractals usage in domains ranging from multi-coloured, multi-patterned fractal landscapes of natural geographic entities, image compression to even modelling of molecular ecosystems. Fractal geometry provides a basis for modelling the infinite detail found in nature. Fractals contain their scale down, rotate and skew replicas embedded in them. Of the many different types of fractals that have come into limelight since their origin, the fractals like the Koch curve, the Cantor Set, the Fern leaf, the Sierpinski gasket have eluded both mathematicians and computer scientists alike. And the two-dimensional (2D), 3D, etc. versions of the same have been realized based on the starting axioms/generators. This paper explains the generation of 3D versions of the above mentioned fractals that gives a real-world look and feel in the world of fractal images.***

Keywords— Modeling, Rendering, IFS

1. INTRODUCTION

A fractal is a rough or fragmented geometric shape that can be subdivided into parts, each of which is (at least approximately) a reduced size copy of the whole or in other words, is self-similar when compared with respect to the original shape[2]. The term was coined by Benoit Mandelbrot in 1975 and was derived from the Latin word "fractus" meaning "broken" or "fractional". The primary characteristic properties of fractals are self-similarity, scale invariance and general irregularity in shape due to which they tend to have a significant detail even after magnification-the more the magnification the more the detail. In most cases, a fractal can be generated by a repeating pattern constructed by a recursive or iterative process. Natural fractals possess statistical self-similarity whereas regular fractals such as Sierpinski Triangle, Sierpinski Gasket, Cantor set or Koch curve contain exact self-similarity. This paper presents the generation of 3D versions of the best-known fractals – the Koch snowflake, the Sierpinski gasket and the Cantor set and using the deterministic method of IFS (Function Systems)[3] and affine transformations. The rendering of the same was done in 3D. The programs were implemented in C++. The displayed output of the same based on a typical set of inputs using multiple test cases varied by number of iterations and a given parameter that corresponds to a coefficient value, are presented. The remainder of the paper is organized as follows: Section (2) focuses on methods of recursively generating the 3D images of the fractals –the Koch snowflake, the Sierpinski gasket and the Cantor set. Section (3) shows the experimental results along with the IFS code and the algorithm and Section (4) provides the concluding remarks.

2. METHODS OF GENERATION PAGE SIZE

The method of generation of the 3D fractal image for the three fractals mentioned in this paper is a step-by-step procedure for the three fractals starting with the 2D image, described as follows:

1. Adding a new depth property that is input-dependent to a chosen image. The default depth is chosen to be 3.
2. The generating IFS consists of a (x,y,z) transformation that consists of an (x, y) rotation and a z-based zoom or the scaling factor.
3. Setting the reference frame by specifying a (height, width) pair to the enclosing (movable) window.
4. Using the IFS, auto-generate the self-similar 3D fractal that is within the boundaries of the frame-of-reference.

The dynamic variables involved in this method are:

1. The depth property that **represents an additional new variant that defines a projection**

2. The reference frame itself
3. The (x,y,z) triple representing the translation for the 3D fractal. Here z represents the zoom i.e., scaling factor

The dynamic translation is achieved by auto-capturing the variations in the scaling (the z-variable) value and the corresponding (x,y) rotation; and auto-adapting the GL projection(s) to the conforming viewports.
The program runs as a Windows console application using the Open Glut API, with the 3D fractal image being generated as an IFS-based recursive version of the 2D image. The resulting fractal image is obtained by interactive mouse-based positional translation and context-aware zoom-based scaling in a dynamic fashion with the depth property of 3. Using MATHLAB 3D Image Rendering software, the displayed graphics are processed for 'true' GUI compatibility.

The individual description of each of the three fractals is given in the subsections that follow.

2.1 Koch Snowflake

Start with a cube, apply a 3×3 grid to each face to divide it up into 27 smaller cubes, and throw away the eight corner-pieces. Then do the same thing for each of the smaller remaining cubes, and repeat. The results are given in section 3.1 below.

2.2 3D Sierpinski gasket

The Sierpinski gasket is a fractal produced by starting with one big filled-in triangle and dividing the triangle into four smaller triangles, then cutting out smaller and smaller ones from it in appropriate places in a recursion process which involves blanking out the center triangle and carrying out the above recursion process to operate upon the peripheral triangles. This involves a starting figure, a generator (the transformation) and a recursive process of applying the generator to the starting figure. The Sierpinski Gasket can also be generated by starting with a rectangle as the starting figure and applying a set of affine transformations (a rotation, a scaling and a translation) in each iteration step. The resulting fractal is the Sierpinski Gasket, a self-similar, equilaterally shaped triangular image that is symmetric with respect to the starting triangle. The resulting 3D fractal is obtained by adding a new depth property that is input-dependent and recursive re-generation with the same depth property[1]. The results are given in section 3.2 below.

2.3 *Cantor Set*

The Cantor set is one of the most remarkable fractal created by repeatedly deleting the open middle thirds of a set of line segments. One starts by deleting the open middle third ($\frac{1}{3}$, $\frac{2}{3}$) from the interval [0, 1], leaving two line segments: [0, $\frac{1}{3}$] ∪ [$\frac{2}{3}$, 1]. Next, the open middle third of each of these remaining segments is deleted, leaving four line segments: [0, $\frac{1}{9}$] ∪ [$\frac{2}{9}$, $\frac{1}{3}$] ∪ [$\frac{2}{3}$, $\frac{7}{9}$] ∪ [$\frac{8}{9}$, 1]. This process is continued ad infinitum, where the *n*th set

$$C_n = \frac{C_{n-1}}{3} \cup \left(\frac{2}{3} + \frac{C_{n-1}}{3}\right).$$

The Cantor set contains all points in the interval [0, 1] that are not deleted at any step in this infinite process. The results are given in section 3.3 below.

3. EXPERIMENTALLY GENERATED FRACTAL FIGURES

3.1 *3D Koch Snowflake*

The figures 1 to 4 illustrate the recursive generation by applying the IFS iterations of the 3D Koch snowflake starting from the initiator to the final 3D fractal image.

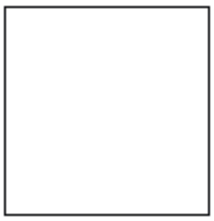

Figure 1: INITIATOR for 3D Koch Snowflake

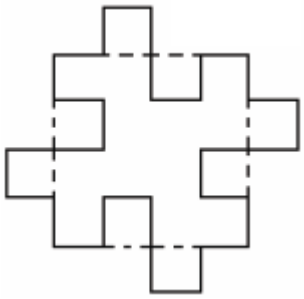

Figure 2: KOCH Snowflake after applying IFS iterations

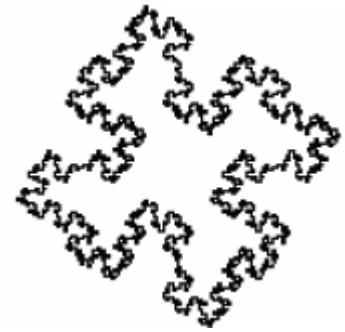

Figure 3: KOCH Snowflake after further IFS iterations

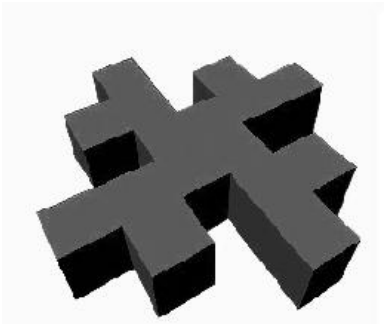

Figure 4: Final 3D Koch Snowflake

3.2 *3D Sierpinski gasket*

The figures 5 to 11 illustrate the recursive generation by applying the IFS iterations of the 3D Sierpinski gasket starting from the initiator to the final 3D fractal image.

Figure 5: INITIATOR for Sierpinski Gasket

Figure 6: SIERPINSKI Gasket after applying IFS iterations

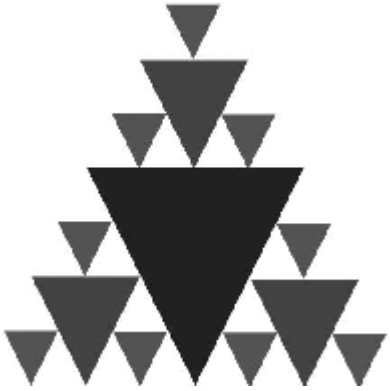

Figure 7: SIERPINSKI Gasket after applying further IFS iterations

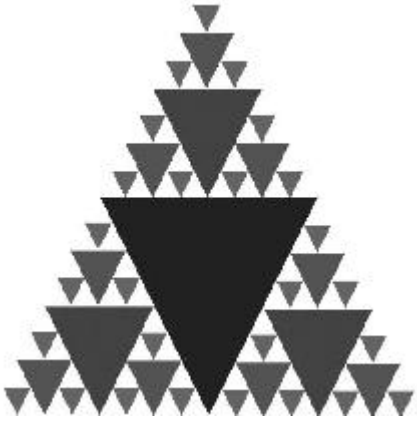

Figure 8: SIERPINSKI Gasket after applying further IFS iterations

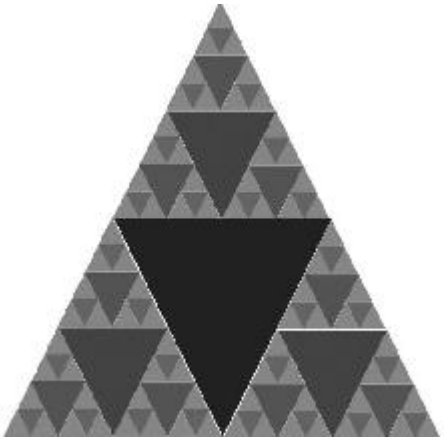

Figure 9: SIERPINSKI Gasket after applying further IFS iterations

Figure 10: 3D Sierpinski Gasket after applying IFS iterations

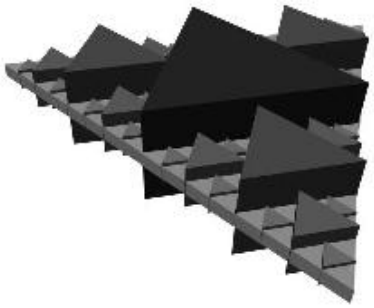

Figure 11: FINAL 3D Sierpinski gasket

3.3 *Cantor Set*

The figures 12 to 19 illustrate the recursive generation by applying the IFS iterations of the 3D Cantor Set starting from the initiator to the final 3D fractal image.

Figure 12: INITIATOR for Cantor Set

Figure 13: CANTOR Set after initial IFS iterations

Figure 14: CANTOR Set after applying further IFS iterations

Figure 15: CANTOR Set after applying further IFS iterations

Figure 16: 3D Cantor Set after applying further IFS iterations

Figure 17: FINAL 3D Cantor Set

4. CONCLUDING REMARKS

This paper explained the generation of 3D versions of the Koch Snowflake, the Sierpinski gasket and the Cantor set fractals the rendering of which gives a real-world look and feel in the world of fractal images. The two-dimensional (2D), 3D, etc. versions of the same have been realized based on the starting axioms/generators. In making further research on fractals[4], these fractals help us to explore the science of fractals in generation of new fractals based on the ideas presented here.

5. REFERENCES

[1] Bulusu Rama, Jibitesh Mishra, "Using 3D Sierpinski gasket to generate and recursively re-generate 3D fractals – Closing the self-similarity loop", International Journal of Graphics, Vision and Image Processing, vol. 12, no. 1, pp.43-48, 2012.
[2] Benoit B. Mandelbrot, The Fractal Geometry of Nature, W.H. Freeman and Company, USA, 1982.
[3] Andrew Top, "3D Iterated Function Systems", http://www.andrewtop.com/Projects/IFS3D, 2007.
[4] Eric Baird, "Alt. Fractals: A visual guide to fractal geometry and design", Chocolate Tree Books, UK, 2011