Indian Journal of Science and Technology, *Vol 12(14), DOI: 10.17485/ijst/2019/v12i14/143038, April 2019*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

The Study of Closedness and Self-similarity of 3D Fractals

Bulusu Rama[1], S.K. Khaja Shareef[1], T. Nirmala[1] and J. Thirupathi[2]

[1]Department of Computer Science and Engineering, MLR Institute of Technology, Dundigal – 500043, Hyderabad, Telangana, India; bulusurama1967@gmail.com, khaja.sk08@gmail.com, nirmala99teegala@gmail.com
[2]Department of Computer Science and Engineering, Institute of Aeronautical Engineering, Dundigal – 500043, Hyderabad, Telangana, India; j.thirupathi@iare.ac.in

Abstract

Objective: To illustrate a methodology that demonstrates the closedness and self-similarity of 3D fractals taking 3D Sierpinski Gasket as an example. **Methods/Statistical Analysis:** The 2D and 3D images of the Sierpinski Gasket were perceived by taking the initial generators as a foundation giving rise to either Sierpinski Triangle/Sierpinski Pyramid or Sierpinski Carpet/Sierpinski Gasket. This method ensures that final 3D Sierpinski Gasket is generated which an exact copy of the original image is taken. **Findings:** This is accomplished by taking a cube as the base and apply the same algorithm in a recursive manner by way of IFS-like transformation consisting of ((x,y) rotation, z(zoom)) and changing the depth parameter from 3 to 2 to 1 in that order, giving rise to a new 3D Sierpinski Gasket that has a resemblance to an exact copy of the original 3D cube-based Sierpinski Gasket. The results depict the closedness in self-similarity aspect of fractal images and thereby give a real-time vision of fractal images. **Application/Improvements:** The results embodied from the above method can be useful for further research and the extension of this work gives us more innovative analysis in the field of 3D fractals and the like.

Keywords: 3D Fractals, Recursion, Sierpinski Gasket Closedness, Self-Similarity

1. Introduction

As on today, the science of fractals is entirely a new domain of research in the scientific and technological era with a broad range of applications. Typically, fractals have the features of self-similarity, scale invariance and non-uniformity in the figure there by having a notable detail even though they are magnified – the more the exaggeration, the more the detail. In general, fractals are produced by a reiterated pattern built up recursively. Natural fractals exhibit statistical self-similarity and regular fractals have exact self-similarity.

Taking into account the 3D rendering of Sierpinski fractal, we have two cases:

1. With triangle as the initial axiom/generator, we obtain the Sierpinski Triangle and a set of n-dimensional Sierpinski Pyramid(s) with n>=3.
2. With square as the starting axiom/generator, we get the Sierpinski Carpet and a set of n-dimensional Sierpinski Gasket where n>=3.

In this paper, the authors describe a method that illustrate the property of closing the self-similarity loop of 3D fractals by initially taking the Sierpinski Gasket fractal as the source image with a cube as the starting base line shape. The original C program as in is used in the above implementation. The output obtained from this a representative set of inputs as the basis, is presented.

**Author for correspondence*

As an illustration, the results conforming to the above by using IFS are given in Section 3(A).

2. Methodology that Demonstrates the Closedness and Self-Similarity of 3D Fractals Taking 3D Sierpinski Gasket as an Example

The step-by-step methodology of generating 3D fractals with the cubed version of the 3D Sierpinski Gasket as the basis is given below:

1. A new depth property having input-dependence to the chosen 3D cube image is added (default depth = 3).
2. The IFS generation is as follows- A (x,y,z) transformation consisting of (i) an (x,y) rotation, (ii) a z-based scaling factor (zoom factor).
3. The reference frame is set as follows: A (height, width) parameter tupleto movable window. (Default color scheme is taken to be red/black for each cubed-based 3D fractal.
4. Eight self-similar fractal blocks are auto-generated using the IFS. Then these are positioned at the 8 different corners of its parent cube fractal. As a result, we obtain a linked chain of smaller but self-similar 3D Sierpinski Gaskets which are situated within the boundary frame-of-reference.

This method involves the following dynamic variables:

1. Thedepth property representing an additional variable which varying in nature which conforms to a projection.
2. The reference frame itself.
3. The triple (x, y, z) representing3D fractal translation, where z is the zoom or scaling factor.

The dynamic translation is obtained through the process by auto-capturing the z-variable variations, together with the appropriate (x,y) rotation; and the GL projection is auto-adapted to corresponding view ports.

2.1 The Output

The program executes as windows console application by making use of the Open Glut API. The resultant 3D Sierpinski Gasket is obtained as an IFS recursive version of the 2D Sierpinski Carpet image; the base initiator for the IFS being a cube. The final output is a succession of fractal images which are obtained by means of interactive translation and scaling operations dynamically, and changing the depth from 3 to 2 to 1. The graphics obtained from this are processed for "true" GUI compatibility through the use of MATLAB 3D Rendering software[1-4].

The original 3D Sierpinski Gasket (cubed version) is depicted in that it has a depth = 3. On mouse-over shearing and skewing of the same, they are re-generated, all of them also having a depth = 3.The 3D Sierpinski Gasket of is obtained by applying the above angular shearing to surpass a depth of 2. The final gives the 3D Sierpinski Gasket generated by applying the above angular shearing to imitate a depth of 1.

It demonstrates that the 3D Sierpinski Gasket of depth 3 and depth 1 are almost exactly alike in every detail in geometric similarity – a result that reveals the closedness and self-similarity of 3D fractals, the self-similarity being symmetric in nature[1-2].

Additional details and other related techniques for further research can be found in the book(s) as in[5] and in[6-8].

3. Experimental Results

(A) The results of the above experimental procedure on 3D Sierpinski Gasket which are depicted through Figures 1 to 7.

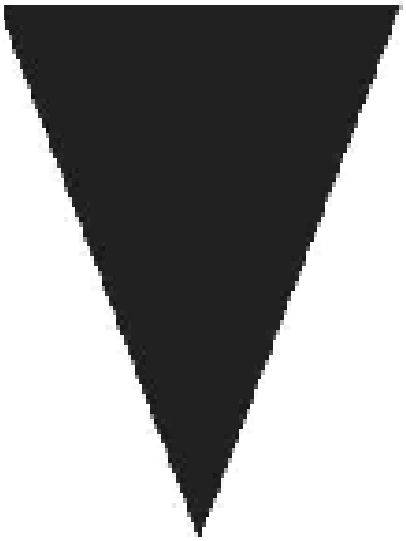

Figure 1. Initiator for Sierpinski gasket.

Figure 2. Sierpinski gasket after applying further IFS iterations.

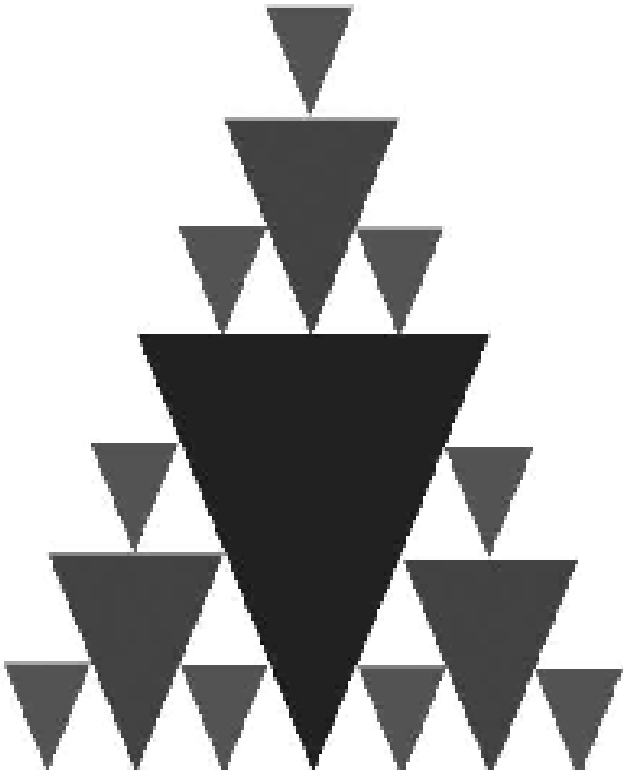

Figure 3. Sierpinski gasket after more IFS iterations.

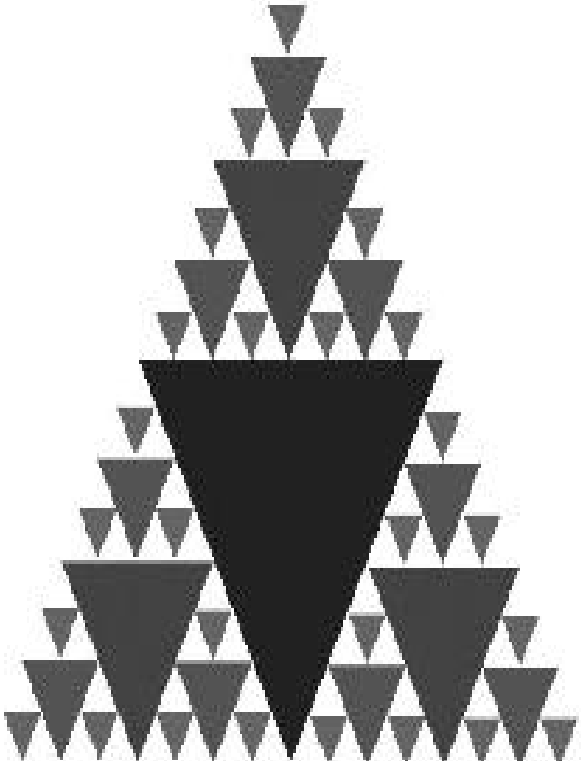

Figure 4. Sierpinski gasket after some IFS iterations.

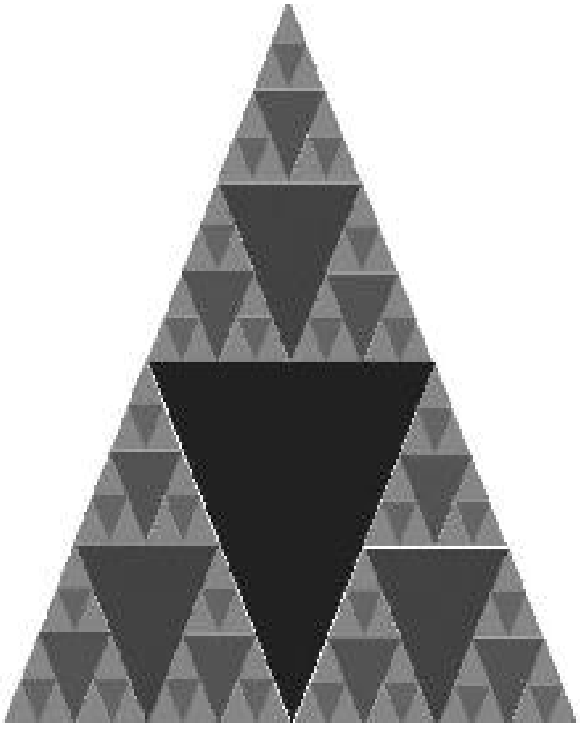

Figure 5. Sierpinski gasket on further iterations.

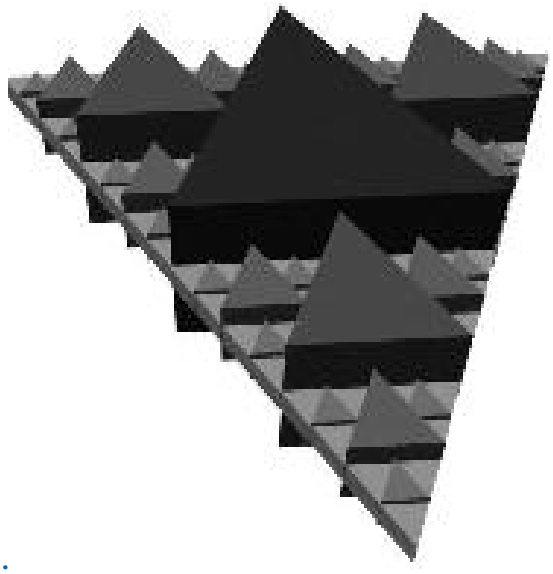

Figure 6. 3D Sierpinski gasket after 5th IFS iterations.

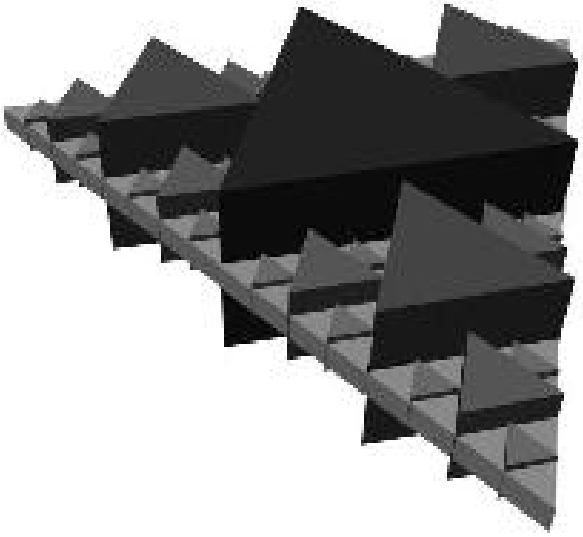

Figure 7. Final 3D sierpinski gasket.

(B) It demonstrates one way of presenting the closedness and self-similarity feature of fractals. The conclusion that can be predicted from this experiment is as follows:

Start with a cube to obtain the 3D Sierpinski Gasket from the 3D Sierpinski Triangle and recursively re-generating the 3D fractal to come out with a new 3D version of Sierpinski Gasket that is almost a 3D 360-degree self-similar image of the original fractal, which can be constructed as a result of the syndicate of the successive iterated version, each of which is a 3D fractal pore of the initial 3D Sierpinski Gasket image. The software used is the C++for the program and the Open Glut Graphics Library for the 3D rendering. The program executes as a windows console application with the input being the depth variable as the first command-line argument (most likely from 1 to 9).

As suggested by the Figures 8-10, the original Sierpinski Gasket uses a square asinitiatorand the IFS generator. The equivalent 3D version of the Sierpinski Gasket Figures (11-13) makes use of a cube as initiator with depth = 3 for all of them; Figures 14-15 and Figure 3 are the maneuvered forms got by recursive re-generation with depth =3. Figure 16 is the form of the Sierpinski Gasket obtained from by recursive re-generation with depth = 2. The final, Figure17 apparently shows the starting 3D Sierpinski Gasket as a 3D cubed version of Figure 18 with depth changed to 1.

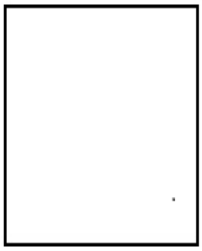

Figure 8. Initial 2D sierpinski gasket – square.

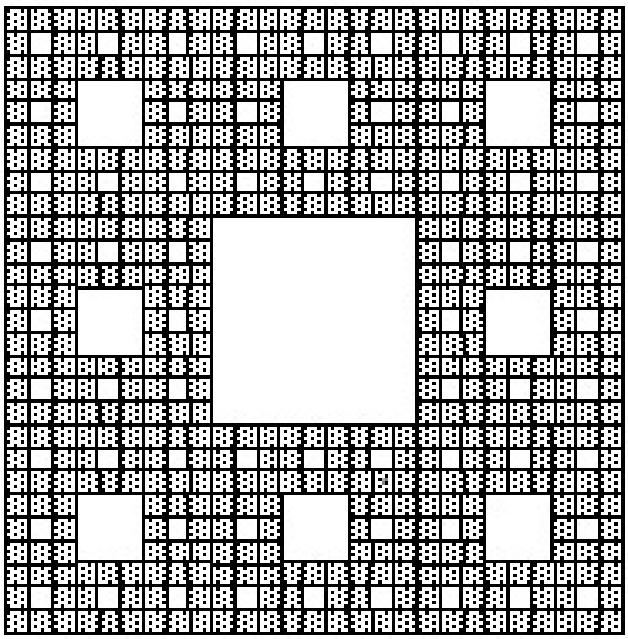

Figure 9. 2D Sierpinski gasket after 3 IFS-iterations applied to Figure 8.

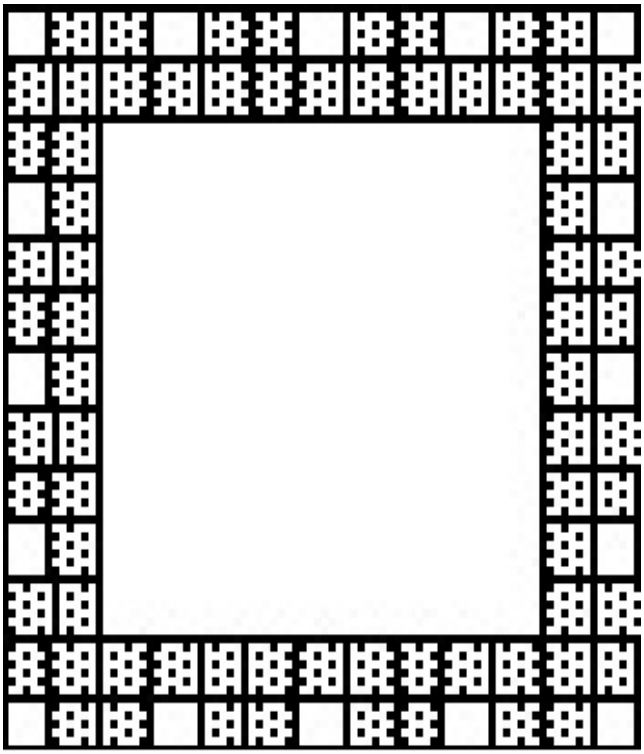

Figure 10. 2D Sierpinski gasket, after applying IFS transformations randomly (to Zoom-in) on Figure 9.

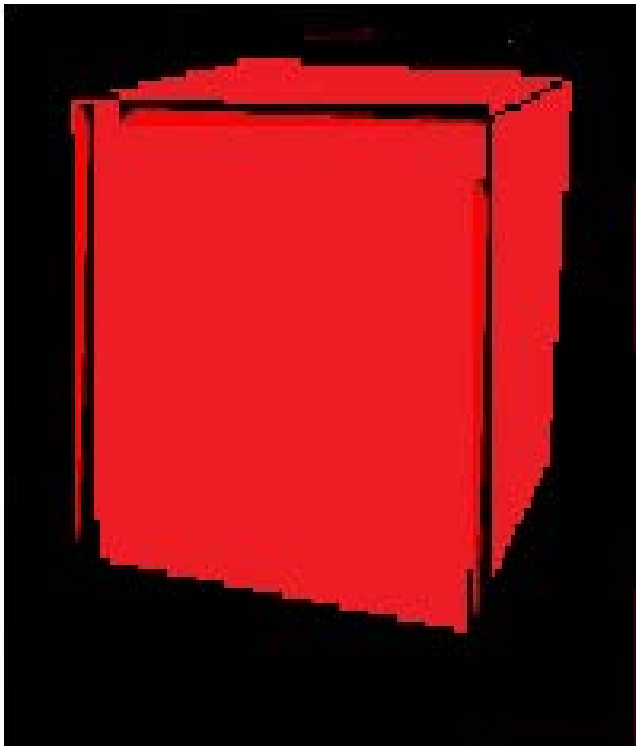

Figure 11. Initiator for the Sierpinski gasket (cube based) having depth=3.

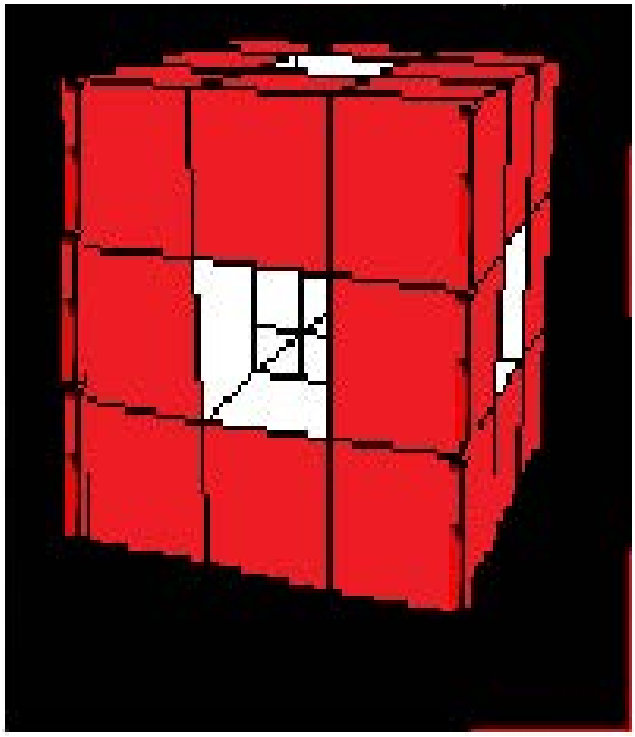

Figure 12. Generator for the Sierpinski Gasket (cube based) having depth=3. The generator is based on the IFS code as outlined in the end of section 3.

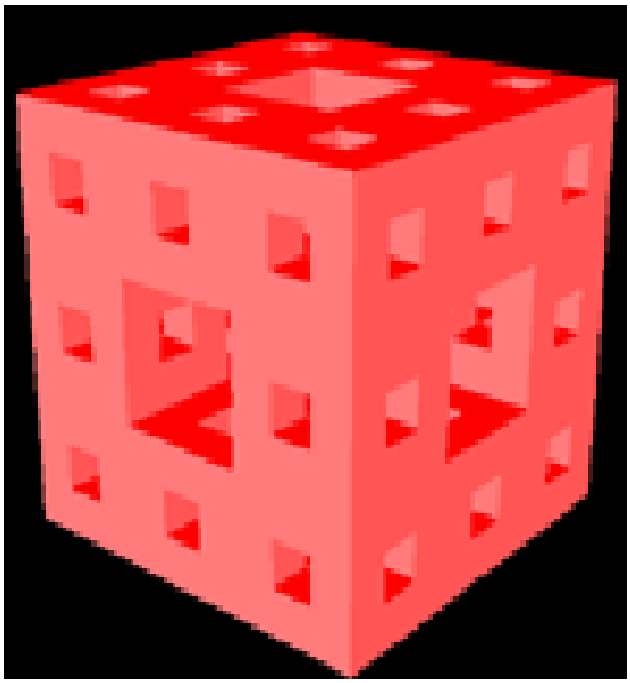

Figure 13. Original 3D Sierpinski gasket (cube-based) having depth=3 obtained from the initiator and generator as shown in Figures 11 and 12.

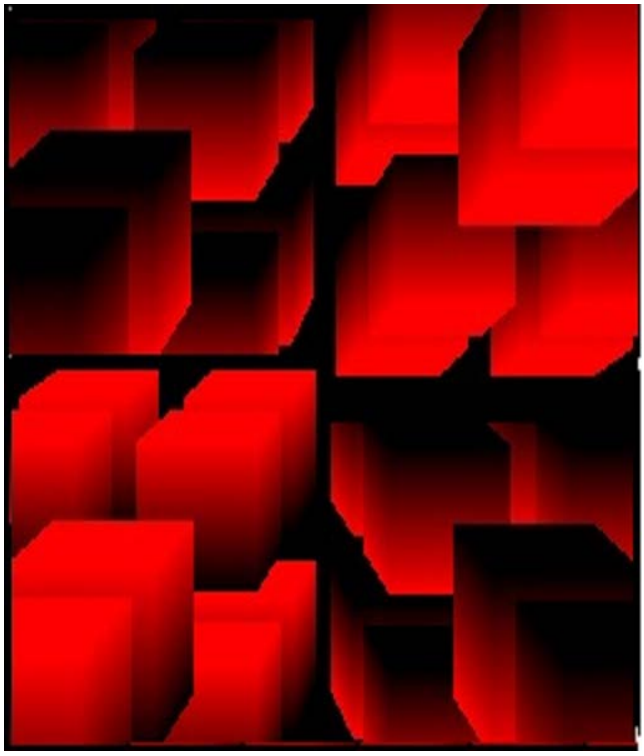

Figure 14. 3D Sierpinski gasket (cube-based) having depth=3 obtained by recursive re-generation of Figure 13 (using the generator as shown in 12).

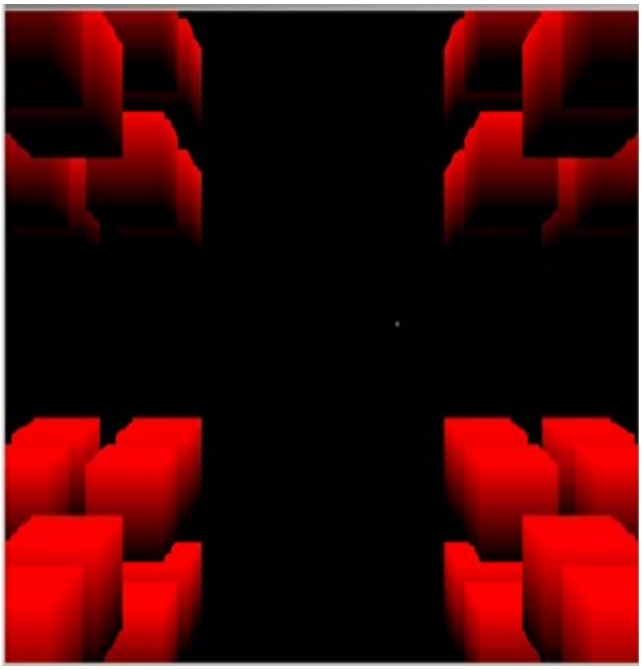

Figure 15. 3D Sierpinski gasket (cube-based) having depth=3 obtained from recursive re-generation of Figure 14 (using the generator as shown in Figure 12).

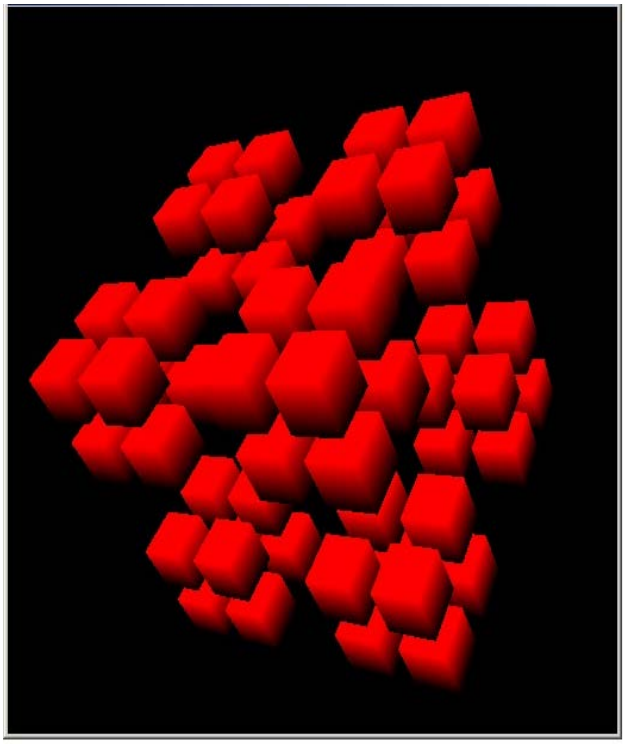

Figure 16. 3D Sierpinski gasket (cube-based) having depth=3 and applying the generator on Figure 15.

Figure 17. 3D Sierpinski gasket (cube-based) generated using depth=2 applying the generator on Figure 16.

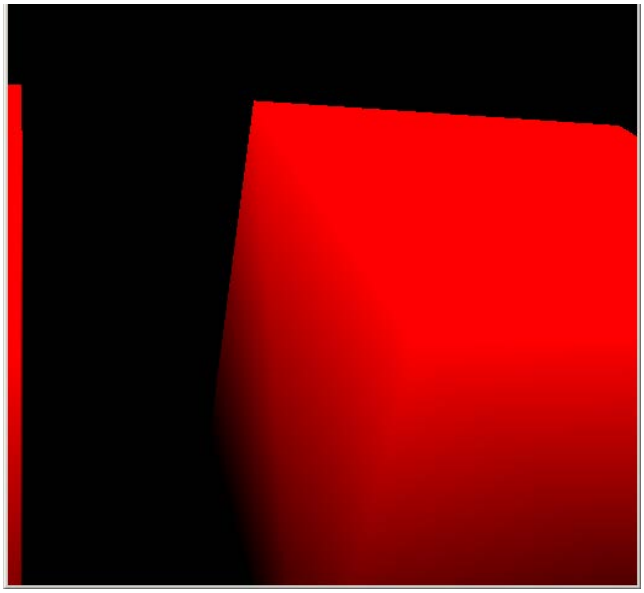

Figure 18. The 3D Sierpinski gasket (cube-based) generated using depth=1 applying the generator on Figure 17.

4. Conclusion

We present a technique of algorithmically maneuvering 3D fractals. It depicts the self-similarity and closedness of 3D fractals by a way of generating a 3D Sierpinski Gasket starting with a cube as the base line shape and recursively applying the same algorithm by way of IFS-like transformations of ((x,y) rotation, z (zoom)), and varying the depth property from 3 to 2 to 1 in that order. The source image of the 3D Gasket is generated based on the 2D version of the same that is achieved on applying IFS transformations to a square figure after n iterations, n>=3. The displayed output based on a classic set of inputs is presented. The results demonstrate the closedness in self-similarity aspect of fractal images and thereby give a real-timevision of fractal images, and how when using theory, computation and experimentation, show that fractal geometry is an art and a science.

5. References

1. New methods in fractal imaging. Date accessed: 26/07/2006. https://ieeexplore.ieee.org/document/1663807.
2. Hart JC, De Fanti TA. Efficient anti-aliasedrendering of 3-D linear fractals. Proceeding SIGGRAPH '91 Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques, 1991; 25(4):91–100. https://doi.org/10.1145/122718.122728.
3. Norton, A. Generation and display of geometric fractals in 3D, Computer Graphics.1982; 16(1):61–67. https://doi.org/10.1145/965145.801263.
4. Van Wijk JJ, Saupe D. Image-based rendering of iterated function systems, Computers and Graphics. 2004; 28(6):937–43. https://doi.org/10.1016/j.cag.2004.08.005.
5. Monro DM, Dudbridge F. Rendering algorithms for deterministic fractals, IEEE Computer Graphics and Application.1995; 272(17):32–41.
6. Generation of 3 Dfractal images for Mandelbrot Set. Date accessed: 01/2011. https://www.researchgate.net/publication/220846428_Generation_of_3D_fractal_images_for_Mandelbrot_set.
7. Rama B, Mishra J. Generation of 3 DFractal Images for Mandelbrot and Julia Sets, Special Issue of International Journal of Computer and Communication Technology. 2010; 1(2-4):178–82.
8. The Fractal Geometry of Nature. Date accessed: 15/08/1982. https://www.amazon.in/Fractal-Geometry-Nature-Benoit-Mandelbrot/dp/0716711869.