Evaluating Second Life for the Collaborative Exploration of 3D Fractals.

Paul Bourke

WASP, University of Western Australia

Abstract

This paper explores the use of the online digital world Second Life as an environment in which one can represent and explore three dimensional fractals, and in addition, present them to others in a collaborative and engaging fashion. Second Life at its core provides a means whereby multiple remote participants can engage with 3D geometry within a virtual environment. It has been chosen as a likely candidate for this exploration for a number of additional reasons, these include the easy to learn user interface, its relatively widespread uptake compared to the alternatives, the availability of the software for all the major operating systems, its non-aggressive social networking foundation, and its scripting capability. The suitability of Second Life will be evaluated through examples. These examples will attempt to create representations of range of the different types of 3D fractals and a discussion of the outcomes will be presented.

Keywords

Fractal geometry, game engine, multiplayer, immersion, Second Life.

1. Introduction

Image based and 2D fractals have generally been studied more than 3D fractals and their generation is supported by a number of software packages, such as FracInt [1]. In contrast, exploration of higher dimensional fractals requires a more sophisticated 3D user interface, more complex visualization techniques, and places higher demands on the capabilities of the graphics hardware. As a result, software that allows one to study 3D fractals is far less common and often written by individual researchers. Alternatively the 3D fractals are created within general purpose rendering/modeling packages in which the fractals are either created manually or by using an internal scripting/programming language to automate what is often an iterative or recursive procedure. These software tools are often only available for a particular computer platform, are often relatively expensive commercial packages, or are geared towards the interactive representation of the fractal geometry for a single user. As a result the process of conveying and sharing a sense of the 3D geometry is relegated to the production of precomputed 2D projections, namely, rendering to images or movies. Even

these 2D projected representations are generally not presented as a real-time collaborative experience but a delayed-in-time file exchange through email and web pages.

By contrast, multiplayer games by their very nature (at least for first person shooters) allow the participant to engage directly and interactively with 3D geometry within a virtual world. They are generally designed for a broad audience and as such are available for a number of operating systems, are easy to install, and have a well designed user interface. Additionally they tend to exploit the capabilities of modern graphics cards to achieve the highest visual quality for a target frame rate.

It is clear that Second Life can be employed for collaboration and social networking [2], that is after all what most activity within Second Life involves. These characteristics mean it has also been explored for collaborative learning experiences [3]. The discussion here then is to determine to what extent Second Life can be employed to represent and convey 3D fractal geometry. Most construction activity within Second Life involves the manual creation of buildings by choosing from the rich set of geometric building blocks provided (boxes, prisms, spheres, etc), this construction occurs by using what is essentially a built-in 3D modeling system. This manual building process can be applied to the iterative/recursive nature of most (but not all) 3D fractals, but fortunately there is also a built-in scripting language that can be used to automatically construct 3D forms. The evaluation then largely consists of determining to what extent can the Linden Scripting Language (LSL) of Second Life [4] be used to create some of the classical 3D fractals. Only actual 3D fractals will be considered, image or geometric fractals that are only 2D can be explored collaboratively with simpler video conferencing tools that generally support image sharing through whiteboards, for example.

2. Evaluation

The first classical 3D fractal to be considered is the Menger sponge, first described by the Austrian mathematician Karl Menger [5]. This is certainly an object that can be created manually quite easily with Second Life modeling tools. One begins with a single cube and duplicates it 27 times in a 3x3x3 grid, then remove the cube in the center of each face and center of the entire grid. This resulting collection of 20 cubes is then grouped together and considered to be the cubic element to which the process is repeated. This replication and positioning when performed manually quickly becomes tiring and can be time consuming as the number of cubes increases. Fortunately it is relatively easy to implement the iterative process using the Second Life scripting language. An example of a LSL script is given below

that takes the current stage of a Menger sponge as input and performs one iteration to create the 20 copies for the next stage.

```
default {
    touch_start(integer anumber) {
        integer i;
        integer j;
        integer k;
        vector p;
        float size = 1; // adjusted for the scale at each stage

        for (k=-1;k<=1;k++) { // “vertical” layer
            for (i=-1;i<=1;i++) {
                for (j=-1;j<=1;j++) {
                    if (k == 0) {
                        if (i != 0 && j != 0) {
                            p = llGetPos() + <-size*i,-size*j,k*size>;
                            llRezObject("MengerLast", p, ZERO_VECTOR, ZERO_ROTATION, 1);
                        }
                    } else if (i != 0 || j != 0) {
                        p = llGetPos() + <-size*i,-size*j,k*size>;
                        llRezObject("MengerLast", p, ZERO_VECTOR, ZERO_ROTATION, 1);
                    }
                }
            }
        }
    }
}
```

Figure 1. Menger sponge.

The exact details of some of the functions and how to create and activate the script are not appropriate here, the purpose is simply to show the C/Java style of the scripting language. In

general terms, a script is activated by touching the object to which the script is attached, the result of applying it once, twice and three times is shown in figure 1.

One characteristic of Second Life and most gaming environments is texture mapping, that is, a means of presenting apparently complex structure by using images rather than geometry. Modern graphics cards are optimized to handle these texture maps and Second Life exploits this capability. In figure 1 note how additional apparent stages have been conveyed with a texture map, so while figure 1 shows four geometric stages, an additional three stages are suggested by the texture map.

Within the Second Life environment the fractal structures discussed here are represented as physical objects within a virtual world, participants can enter the cavities (but not the solid portions), fly in and around, all the time potentially engaging in discussions by text or talking to other participants. This collaborative exploration of fractal geometry within a virtual world is a truly exciting experience for anyone who has tried to share a sense of such forms using traditional static 2D rendered images.

This first example highlights a significant limitation of Second Life for the representation of many fractals, that is, the limited number of geometric primitives supported within a particular area of land. The number of cubes within the Menger sponge increases by a factor of 20 on each iteration so it is a good example of a fractal that, if represented geometrically, will eventually consume any graphics system. The above 8000 cubes would be impossible to create in most areas within Second Life, indeed more than 1000 geometric primitives is usually problematic. These imposed limits are not due to the power of current graphics card technology but rather pragmatic considerations the Second Life developers need to consider in order to prevent activities within the virtual environment that would result in some participants having a less than satisfactory experience. A secondary constraint is a limit of 256 geometric elements that can be linked (grouped) together to form a single entity. While not an impediment to the creation of complex objects, it does place constraints on the management of fractals made from a large number of elements.

Chaotic attractors in 3D are often represented as 2D projections using many thousands, if not millions of points on the projection plane. The sense of such attractors in 3D can still be conveyed with a significantly lower number of primitives and with careful design. The classical Lorenz attractor is described by the following three differential equations

$$dx / dt = 10 (y - x) \qquad dy / dt = x (28 - z) – y \qquad dz / dt = xy - 8 z / 3$$

Figure 2 is one possible representation of the Lorenz attractor in 3D within Second Life. It may not be represented with the same fidelity as other static 2D representations but it still

conveys the key features. Second Life is unlikely to be suitable for real time or animated fractal systems, primarily because any change in the geometry requires a transaction with the remote server in order for the other remote participants to be able to experience the change. The script for the Lorenz attractor creates cylinders one at a time between adjacent points on the attractor, there are typically only a few cylinders created per second (depending on the available bandwidth between the client and the server). This serves to provide instructive clues as to the behavior of the chaotic system, one can observe paths that seem to be coincident for a few cycles only to eventually see them suddenly diverge.

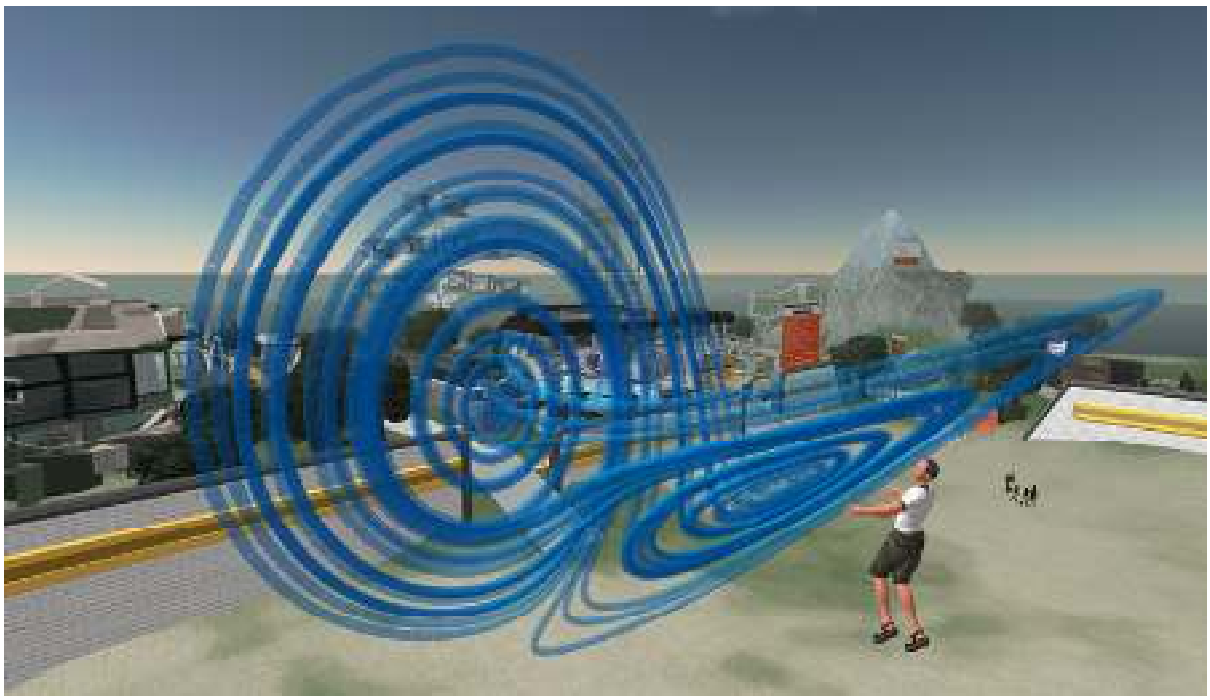

Figure 2. Lorenz attractor.

The texturing ability that was illustrated in figure 1 to give a higher apparent iteration depth can be employed to convey a sense of some fractals that do require many more geometric primitives than could ever be represented within Second Life. In figure 3 the visual field from within a 3D DLA (Diffusion Limited Aggregation) simulation [6] containing many hundreds of thousands of spheres and cylinders is rendered as a full high resolution 360x180 degree spherical projection. This image can be applied as a texture to a single sphere in Second Life and if the participant is located in the center of that sphere he/she can look around and experience what it would be like being inside the DLA root system. This simple technique can result in a very powerful sensation of 3D geometry even though it is no more than a texture mapped onto a sphere.

Second Life does place a limit of 1024 pixels on the size of a texture, larger textures such as the 4096x2048 texture created here are subsampled automatically when uploaded from the client software to the server. In order to achieve a higher quality result the spherical projection can be split into multiple smaller textures, each of which is applied to a section of a sphere. This texture limit is more of an inconvenience and understandable precaution given that there are variable limits on maximum texture sizes imposed by graphics cards.

Figure 3. A 3D diffusion limited aggregation model represented as a spherical projection onto a single sphere.

There exist a class of fractals in 3 dimensions that are best computed as a volumetric dataset, that is, at each small rectangular bounded region (voxel) within a confined volume some scalar value is known. In the case of some 3D and 4D Mandelbrot style fractals, the value stored at each voxel is whether and/or how fast a series diverges to infinity. The process of visualizing such structures is called volume rendering and is commonly performed by taking a number of parallel planar slices through the volume to form a collection of 2D images. The colour at each pixel within the images are defined by mapping voxel values to colour and transparency, carefully chosen so as to reveal the internal structure within the volume. These images are then applied as textures to an equivalent number of planes within a 3D rendering package. This form of volume visualization can be implemented in Second Life, the textures are uploaded through the Second Life application interface after which a LSL script

automatically creates the planes and assigns the correct textures to each. The example in figure 4 is a four dimensional fractal [7] describing the behavior of the series

$$z_{n+1} = (z_n + c_1)^2 + c_2$$

where z_n, c_1, and c_2 are all complex and $z_o = 0+0i$. Figure 4 is the slice $Re(c_1)=0$ through this 4D fractal resulting in a 3D solid object. The red region is where the values of $Im(c_1)$, $Re(c_2)$, and $Im(c_s)$ cause the series to tend to zero, the semitransparent blue region are the points where the series escape to infinity. The main inconvenience to this process is the time required to upload, in this example, a 100 textures from the client machine to the Second Life server.

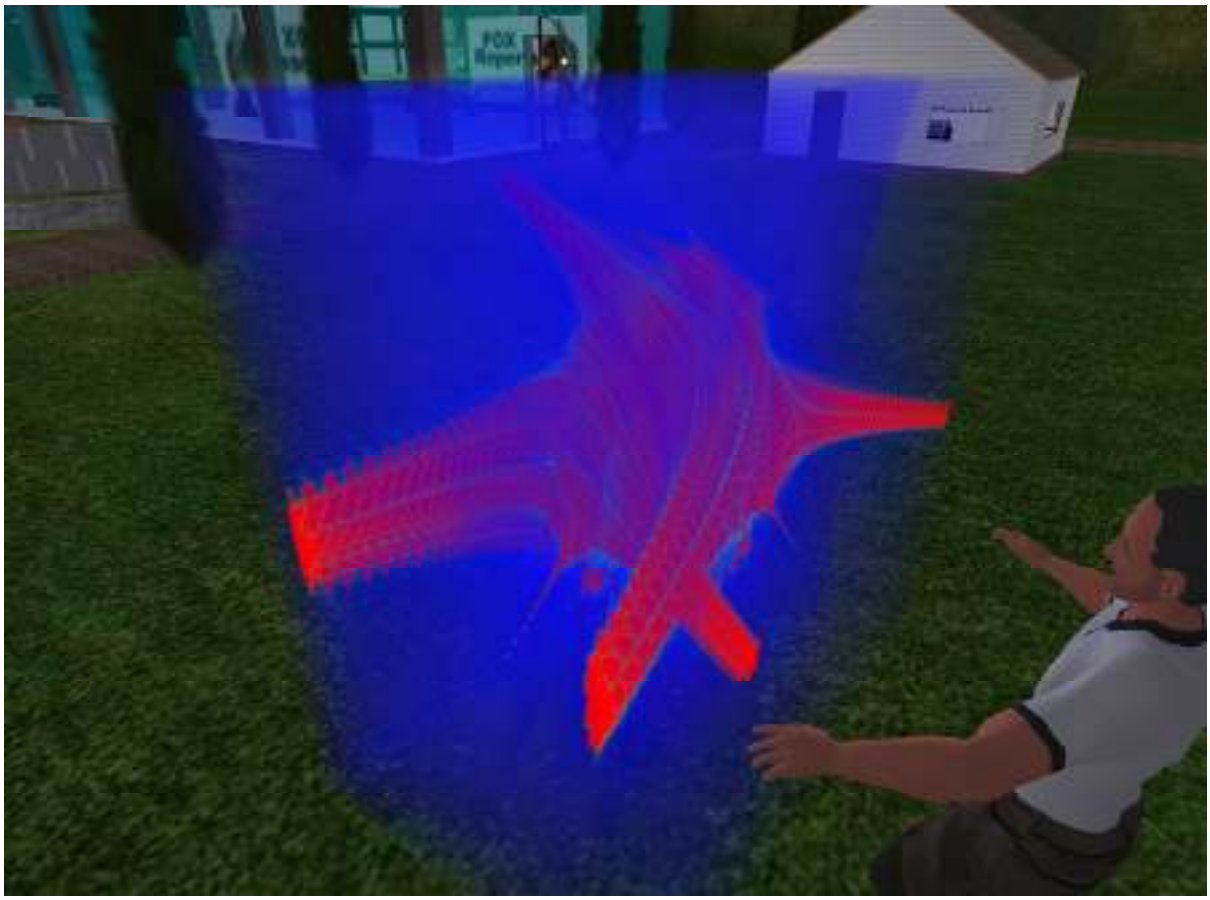

Figure 4. A planar slice through a 4D fractal resulting in a 3D volumetric dataset.

As a final example, there are fractal forms that may not be able to be computed efficiently or at all with a LSL script. In these cases the geometry of the fractal can be created in software external to Second Life, a representation of the geometry can then be created using a LSL script that reads the externally generated data. One way to format data for convenient use by

LSL scripts is to use a LSL “list” data structure. The following is an example of how a number of sphere positions and radii may be represented as a LSL list.

```
list spherelist = [
 <0,0,1>, 1.5,
 <0.5,0,0.5>, 0.5,
 etc ...
];
```

Such a list could be used to describe an Apollony fractal [8]. Software external to Second Life computes the positions and radii of each sphere and saves this data in this list structure in a plain text file. This text file can then be copy/pasted into Second Life and acted on by a script. As an example see the result in figure 5 comprising of an approximation to the Apollony fractal consisting of 500 spheres. In this case the script reads each item in the list and instantiates a sphere in the virtual world at the correct position and with the appropriate radius. The script may in addition assign colour, transparency, and texture to each sphere. There are limits to the number of entries in a list but this is little more than an inconvenience since a larger numbers of entities can be split into a number of smaller individual lists.

Figure 5. Apollony fractal generated by importing externally generated data.

Conclusion

While Second Life provides a cross platform and highly engaging virtual environment that can represent many 3D fractals, there are clear limitations. It should be pointed out that an attempt is being made to use software for a task it was not necessarily designed to perform. In

most cases the limitations identified exist for pragmatic reasons and are intentional in order to give a satisfactory user experience across a range of hardware and network capabilities. The strengths of Second life revolve around the collaborative aspects that are unavailable in any of the existing 3D fractal generation tools. The key limitation for fractal representation is the low number of geometric primitives supported per land area but the effect of this can be minimized with careful design. If the fractal can be created within these limitations then Second Life as collaborative experience and a means of presenting 3D fractals in an engaging way is perhaps unsurpassed.

Acknowledgement to Bowling Green State University for providing a sand pit that supports a large number of primitives. Much of the exploration here would have otherwise been less efficient.

References

[1] Fracint, Web Reference. [http://spanky.triumf.ca/www/fractint/fractint.html]

[2] Livingstone, D., Kemp, J., Massively multilearner: recent advances in 3D social environments. Computing and Information Systems Journal (2006), School of Computing, University of Paisley 10 (2).

[3] Antonacci, D.M., Modaress, N., Second Life: The Educational Possibilities of a Massively Multiplayer Virtual World (MMVW). EDUCAUSE Western Regional Conference, April 26, 2005, San Francisco.

[4] Linden Scripting Language, Web Reference. [http://secondlife.com/whatis/scripting.php]

[5] Menger, K., General Spaces and Cartesian Spaces, Communications to the Amsterdam Academy of Sciences (1926). English translation reprinted in Classics on Fractals, Gerald A.Edgar, editor, Addison-Wesley (1993).

[6] Bourke, P.D., Constrained diffusion limited aggregation in 3 dimensions. Computers and Graphics, Volume 30, Issue 4, August 2006, Pages 646-649.

[7] Danca, M.F., Tang, W., and Chen, G., A switching scheme for synthesizing attractors of dissipative chaotic systems, Applied Mathematics and Computations, (2007).

[8] Bourke, P.D., Apollony fractal. Computers and Graphics, Vol 30, Issue 1, January 2006, pages 134-136.