Computers & Graphics 33 (2009) 113–117

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Chaos and Graphics

Evaluating Second Life for the collaborative exploration of 3D fractals

Paul Bourke *

WASP, University of Western Australia, Perth, WA 6009, Australia

ARTICLE INFO

Keywords:
Fractal geometry
Game engine
Multiplayer
Immersion
Second Life

ABSTRACT

This paper explores the use of online digital world Second Life as an environment in which one can represent and explore three-dimensional (3D) 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.

1. Introduction

Two-dimensional (2D) fractals have generally been studied more than three-dimensional (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 capabilities of the graphics hardware. As a result, softwares that allow one to study 3D fractals are far less common and often written by individual researchers. Alternatively, 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 available only for a particular computer platform, are often relatively expensive commercial packages, or are geared towards an interactive representation of fractal geometries 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 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 participants 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.

Second Life is an online 3D virtual environment managed with a server–client software model and created by Linden Labs [2]. In addition to participants being able to create and modify their assets in virtual environments, Second Life provides a rich environment for social networking [3]. Indeed this is perhaps the most engaging activity for many participants. These characteristics mean it has also been explored for collaborative learning experiences [4]. The discussion here then is to determine to what extent Second Life can be employed to represent and convey 3D fractal geometries. Most construction activities within Second Life involve 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 the Linden Scripting Language (LSL) of Second Life [5] can 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

* Tel.: +61 8 6488 8097; fax: +61 8 6488 8088.
E-mail address: paul.bourke@uwa.edu.au

0097-8493/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cag.2008.08.004

Menger [6]. This is certainly an object that can be created manually quite easily with Second Life modeling tools. One begins with a single cube, duplicates it 27 times in a $3 \times 3 \times 3$ grid, then removes 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 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);
          }
        }
      }
    }
  }
}
```

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, the above 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 Fig. 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 Fig. 1 note how additional apparent iterations have been conveyed with a texture map, in this case a 2D image of the side of a Menger sponge. Fig. 1 shows four geometric iterations, 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 by default not the solid portions) and 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 can be an exciting and engaging 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.

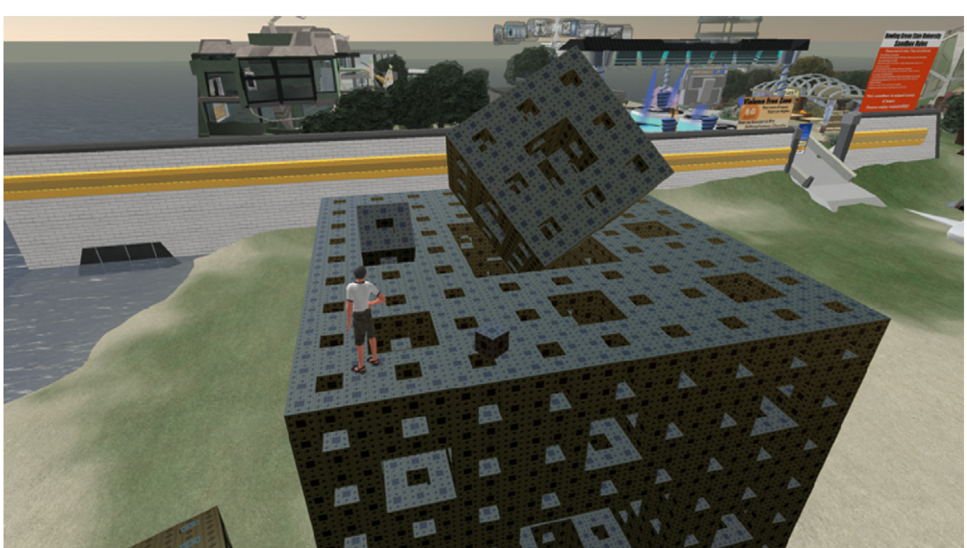

Fig. 1. Menger sponge: four geometric iterations and additional iterations conveyed using texture maps.