International Journal of Computer Science and Information Technology
ISSN: 3005-9682 (Print), ISSN: 3005-7140 (Online) | Volume 2, Number 1, Year 2024
DOI: https://doi.org/10.62051/ijcsit.v2n1.16
Journal homepage: https://wepub.org/index.php/IJCSIT/index

An Overview of Procedurally Generating Virtual Environments

Yinghu Liu

Software Engineering, Chengdu University of Information Technology, Chengdu, China

ABSTRACT

Procedural Generation is a method of using computers to generate content, allowing users to manually create content for virtual worlds is a very time consuming and laborious task, with the development of computer hardware, Procedural Generation is a good alternative, Procedural Generation content can be used as models or sketches to provide inspiration for the artist or to improve efficiency, this paper focuses on the representative algorithms used in Procedural Generation, which are commonly used to produce architecture, plants and terrain.

KEYWORDS

Procedural generation; Parallax Mapping; Skeleton Algorithm; Shape Grammer; Tree-Map Algorithm; L-system

1. INTRODUCTION

In recent years, virtual worlds have been used in the film and video game industry, digital cultural heritage, simulation of urban phenomena and interior and exterior design planning. As virtual and augmented reality applications continue to evolve, there is an increasing demand for methods that enable fast, detailed and interactive 3D content. Hand-built virtual worlds are very rigorous and cannot be easily modified once they are finished, especially for some large-scale scenarios.

Procedural generation is an active research topic, the main goal is to be able to automatically create the desired object with minimal input, procedurally generated objects are also very easy to modify twice, it is used in a variety of fields, such as textures, plants, terrain, buildings, rivers and various artistic creations, this paper will introduce the generation methods for the most frequently procedurally generated content such as plants, buildings, terrain.

2. PROCEDURAL GENERATION OF VEGETATION

The phenomenon of self-similarity is widespread in nature, and fractal structures are more stable and fault-tolerant than Euclidean geometry, with greater determinism, inspired by which was introduced into the field of computer graphics known as L-system [2], a context-independent band grammar that generates each rule that applies to only one symbol in the geometry, starting from the initial structure. Other symbols are not affected by the rules. A new object is formed by replacing some parts of the representation and iteratively applying some rules for string symbols to create branching structures. If a character is treated as an element of a language, then a set of rewrite rules might look like this:

$$b \rightarrow a \tag{1}$$

$$a \rightarrow ab \tag{2}$$

This principle was originally used by Chomsky et al [1] in their description of a programming language.L-Systems requires that each rewrite rule be applied once in each round because plant growth is based on cell division and all cells occur in parallel. If "a" in the above example is used as the initial string, the process can be represented as shown in Figure 1.

a
ab
aba
abaab
abaababa
abaababababaab
...

Figure 1. String generation process

In the L-system, this process describes how plants grow. In more common representations, we mark any substitutions that may occur with parentheses around the corresponding substrings. For example, an L-system can be defined as follows (n is the number of times the rewrite rule is applied, 'F' in the second line is the initial string, and δ is the degree of bending):

$$n = 1, \delta = 25° \tag{3}$$

$$F \tag{4}$$

$$F \rightarrow F[+F]F[-F]F \tag{5}$$

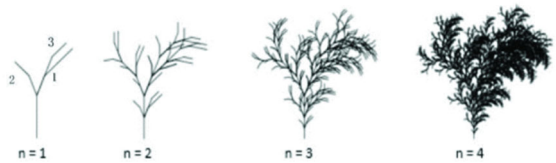

Figure 2. L-system generation results process

The L-system makes it simple to generate plants with different shapes, and if 3D plants need to be generated, simply replace the initial left- and right-turn operations with movements in 3D space at each decision point [1], making the L-system by far the most widely used and successful method for procedurally generating plant content.

3. PROCEDURAL GENERATION OF BUILDINGS

There are a large number of research papers on the development of procedurally generated buildings [3-6], but only a small number of concrete implementations [7 8].Procedural generation has been a

popular research area since 2000.Procedurally generated buildings are generally divided into two parts.

1. the design of the house layout in the floor.

2. building shape and facade generation.

These two together generate a complete building.

3.1. Procedural Layout Generation

Johnson et al. improved the Treemap [9] algorithm and proposed the Squarified Treemap algorithm [10] for filling rooms in a given rectangular building region.

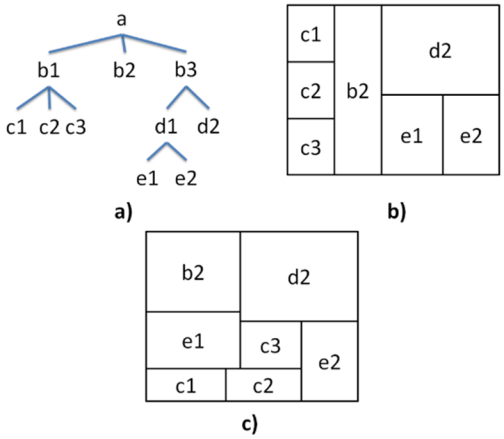

Figure 3. Layout division results

As shown in Fig. 3, compared to the result Fig. 3(b) of the TreeMap algorithm, the result Fig. 3(c) produced by Squarified Treemap is much better in terms of the width to height ratio of the rooms, the rooms should be placed so that the width to height ratio of each room is as close to 1 as possible, to avoid generating rooms that are too long or narrow, but this result lacks utility in the sense that each room can only be connected to another individual room and the whole generated result has no corridor.Mirahmadi and Shami et al. proposed an improvement to the Squarified Treemap algorithm [22], which is able to find corridor paths connecting all individual rooms in a building.

Another limitation of the tree diagram algorithm is that it can only process rectangular regions.The constrained growth method for procedural floorplan generation proposed by Lopes et al [11] is able to process arbitrarily shaped regions.As shown in Fig. 4, the target region is first tiled.For each room, a cell is selected as the starting point, and it grows sequentially in a single direction until it reaches its predefined size.The next iteration is performed in the second iteration, in which a cell is assigned to a room connected to it. Then the next iteration is performed, and in the second iteration, each unfilled cell is assigned to a room connected to it.

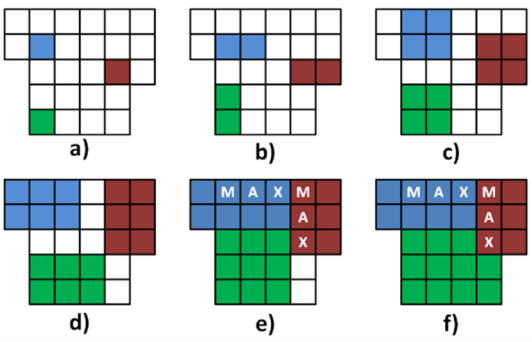

Figure 4. Constrained growth method approach layout

This method allows you to fill circles or triangles by adjusting the size of the cell; the smaller the cell, the more accurate the calculation will be.

3.2. Procedural Facade Generation

In order to obtain a more diverse range of building structures, Wonka et al. introduced the concept of split grammars, a formal context-independent grammar [12] for generating building models.Split grammars are similar to L-systems, where the basic elements are shapes rather than symbols.Müller et al. built on this foundation and developed the shape grammar CGA [13], which, unlike split grammars. Shape Grammar uses context-dependent rules that allow splitting and rotating elements, and CGA-based modelling modelling systems can automatically generate a variety of models based on user-written rules (see Fig. 5) Schwarz et al [14] introduced the CGA++ grammar language, which increases the context sensitivity of the CGA shapes, allowing for combinations of multiple buildings to be combined to produce better results.

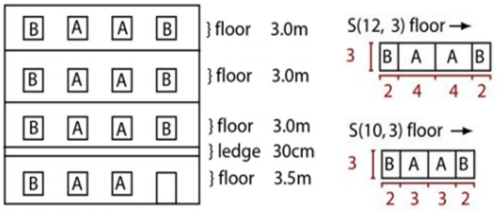

Figure 5. Shape Grammer

Inverse process modelling approaches have also been widely used, Martin et al [15] proposed a guided inverse process modelling approach for analysing existing input building models to obtain a set of shape grammars, and then synthesising new building models based on these shape grammars under the guidance of the user (see Figure 6).

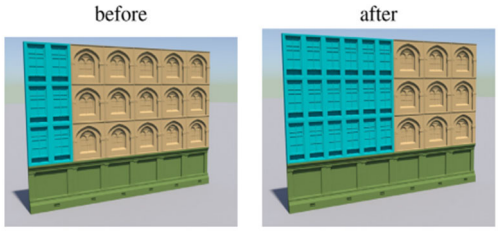

Figure 6. Inverse Precess Modelling

In the field of façade modelling, CityEngine has extended the L-system approach to large-scale urban buildings [16], and has achieved great success in rule-based syntax process modelling, where CityEngine uses a fully parametric, stochastic L-system to extrude the building structure on the basis of an arbitrary building profile, and is able to adapt to global and local constraints.

Finkenzeller et al. in their work [17] provide a fairly comprehensive overview of existing façade modelling techniques, arguing that attention should be paid to the semantic information of each shape in the context of the complete building, and they suggest that this semantic information should be noted in the stylisation rules in order to generate different styles of building façades and apply them to the same building contours.

4. PROCEDURAL TERRAIN GENERATION

Automated terrain modelling is one of the key themes of procedural generation. Its main domain extends from terrain elevation to urban environments. Terrain is generally represented by height maps [18], also known as Digital Terrain Models (DTM). The height of these points is the vertical distance between a terrain point and a reference surface. Typically, a height map consists of a grey bitmap

where the height data is represented by grey shading of bitmap pixels. This is then visualised in 3D space using a polygonal mesh [19]

as shown in Figure 7. The whiter the pixel, the higher the elevation point. Figure 7 shows a grey scale representation of Mount Everest and the surrounding mountains. Higher areas are represented by brighter pixels and lower areas are represented by darker pixels.

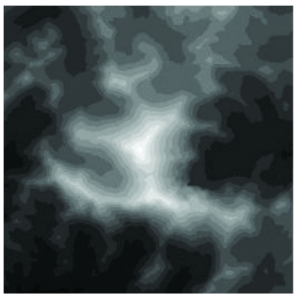

Figure 7. Height map of the Himalayas

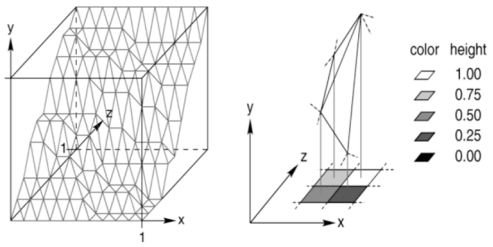

Figure 8. Modifying heights in 3D space based on grey scale maps

Musgrave et al [20], in order to simulate the natural erosion process, in proposed a fractal noise synthesis algorithm that uses noise for local independent control of the generated terrain, which effectively improves the realism of the generated terrain (see Figure 9).

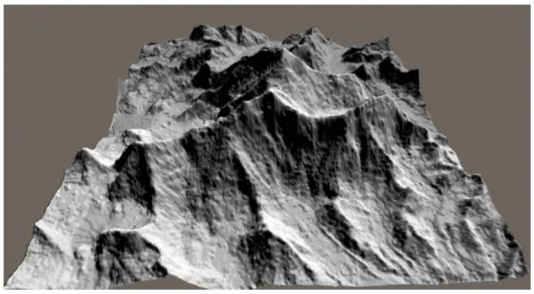

Figure 9. Himalayas modelled by erosion algorithms

Although these erosion simulation algorithms greatly improve the realism of terrain generation, they all require a large amount of computation, another method to simulate nature is the Voronoi Tessellation method [21], Voronoi Tessellation is a natural phenomenon that occurs at mountain range intersections, the Voronoi Tessellation region represents the set of all points closest to the centre of the region, and through mesh mapping, the height of the surrounding points can be adjusted while raising the centre point, although this method can greatly reduce the time spent by the program in generating the terrain, the generation effect is not as good as the former (see Figure 10).

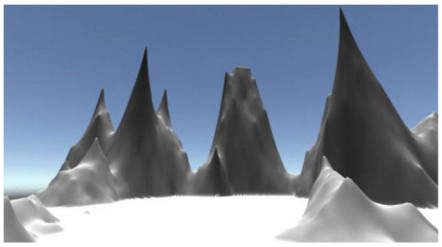

Figure 10. Mountains generated by the Voronoi Tessellation method

5. CONCLUSIONS

Creating content for virtual worlds manually is a very time-consuming and tedious task, procedural content generation can save a lot of time for developers and artists, and also provide artists with new inspirations to stimulate the creativity of designers. PCG has a wide range of application scenarios, it can automate the generation of complex spatial structures such as caves and labyrinths, introduce changes and uncertainties, and provide a unique exploration experience for game players. In game system design, PCG can be used to create diverse enemies, quests, and obstacles to improve game playability. In the film and television industry, procedural techniques can be applied to the development of narrative storylines to generate multi-threaded storylines that enrich film content and enhance audience experience. Audio editing and music creation can also use PCG to generate different styles and rhythms of music, providing unlimited music materials and stimulating the creative energy of musicians. In addition, PCG technology shows great potential in creating large datasets for machine learning, providing diverse and high-quality training materials for algorithm training.

As the field of PCG continues to merge with artificial intelligence technologies, the application of advanced tools such as neural networks and evolutionary algorithms makes PCG smarter and more efficient. They are becoming a hot topic in both research and practice, with more and more researchers devoting themselves to exploring how these algorithms can be used to better meet the precise needs of designers as well as adapt to player preferences. While PCG has made significant strides in automated, personalised content generation, its potential for creating content that meets specific design goals and user expectations remains immense.

REFERENCES

[1] J. Freiknecht and W. Effelsberg, "A survey on the procedural generation of virtual worlds", Multimodal Technol and Interact, vol. 4, no. 4, 2017.

[2] M. H. Tanveer, A. Thomas, X. Wu et al., "Simulate forest trees by integrating l-system and 3d cad files", IEEE 3 rd ICICT , pp. 91-95, 2020.

[3] Brenner, C. Towards Fully Automatic Generation of City Models. 2000. Available online: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.17.4549&rep=rep1&type=pdf

[4] Saldana, M.; Johanson, C. Procedural modeling for rapid-prototyping of multiple building phases. Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. 2013, 5.

[5] Birch, P.J.; Browne, S.P.; Jennings, V.J.; Day, A.M.; Arnold, D.B. Rapid Procedural-modelling of Architectural Structures. In Proceedings of the 2001 Conference on Virtual Reality, Archeology, and Cultural Heritage, Athens, Greece, 28–30 November 2001; ACM: New York, NY, USA, 2001; pp. 187–196.

[6] Martin, J. Algorithmic Beauty of Buildings Methods for Procedural Building Generation. Available online: http://digitalcommons.trinity.edu/cgi/viewcontent.cgi?article=1003&context=compsci_honors

[7] Stocker,J.BuildR2.2003.Available online:http://support.jasperstocker.com/buildr2/

[8] Ibele, T. Building Generator. 2009. Available online: http://tysonibele.com/Main/BuildingGenerator/buildingGen.htm.

[9] Johnson, B.; Shneiderman, B. Tree-maps: A space-filling approach to the visualization of hierarchical information structures. In Proceedings of the 2nd Conference on Visualization’91, San Diego, CA, USA, 22–25 October 1991; pp. 284–291.

[10] artin, J. Algorithmic Beauty of Buildings Methods for Procedural Building Generation. Available online: http://digitalcommons.trinity.edu/cgi/viewcontent.cgi?article=1003&context=compsci_honors

[11] Lopes, R.; Tutenel, T.; Smelik, R.M.; De Kraker, K.J.; Bidarra, R. A Constrained Growth Method for Procedural Floor Plan Generation.

[12] Wonka, P.; Wimmer, M.; Sillion, F.; Ribarsky, W. Instant Architecture; ACM: New York, NY, USA, 2003; Volume 22.

[13] Müller, P.; Wonka, P.; Haegler, S.; Ulmer, A.; Van Gool, L. Procedural modeling of buildings. In Proceedings of the ACM SIGGRAPH 2006, Boston, MA, USA, 30 July–3 August 2006; ACM: New York, NY, USA, 2006; Volume 25, pp. 614–623.

[14] M. Schwarz and P. Müller, "Advanced procedural modeling of architecture", ACM Trans. Graph., vol. 34, no. 4, 2015.

[15] Ignacio Martin and Gustavo Patow. Ruleset-rewriting for procedural modeling of buildings[J]. Computers & Graphics, 2019, 84(C) : 93-102.

[16] G. Kelly and H. McCabe, "Citygen: An interactive system for procedural city generation", Proc. 5th Annu. Int. Conf. Comput. Game Design Technol., pp. 8-16, 2007-Nov.

[17] D. Finkenzeller, Modellierung Komplexer Gebäudefassaden in der Computergraphik, Karlsruhe. Germany:KIT Scientific, 2008.

[18] R. M. Smelik et al., "A Survey of Procedural Methods for Terrain Modelling", 3AMIGAS, vol. 2009, pp. 25-34, June 2009.

[19] R. B. Yehuda and C. Gotsman, "Time/Space tradeoff for polygon mesh rendering", ACM Transaction on Graphics, vol. 15, pp. 12, April 1996.

[20] F. K. Musgrave, C. E. Kolb and R.S. Mace, "The synthesis and rendering of eroded fractal terrains", ACM Siggraph Comput Graph, vol. 23, no. 3, pp. 41-50, 1989.

[21] K. S. Emmanuel, C. Mathuram, A.R. Priyadarshi et al., "A Beginners Guide to Procedural Terrain Modelling Techniques", IEEE 2 nd ICSPC , pp. 212-217, March, 2019.

[22] M. Mirahmadi and A. Shami, "A novel algorithm for real-time procedural generation of building floor plans", Comput. Res. Repository, pp. 7, 2012.