Methods for Procedural Terrain Generation: A Review

Luis Oswaldo Valencia-Rosado(✉) and Oleg Starostenko

Department of Computing, Electronics and Mechatronics,
Universidad de las Américas Puebla, San Andrés Cholula, 72810 Puebla, Mexico
{luis.valencia, oleg.starostenko}@udlap.mx

Abstract. Advances in computer graphics allow to simulate ever growing virtual worlds with a higher level of realism which can even be created in real time. An integral part of these worlds are the terrains which are the physical features of the land. Despite the capabilities of modern computer systems, the creation process still demands high amounts of man-hours. To automatically generate coherent, realistic-looking and useful content is still and open problem, and research focuses on how to automatize these processes while allowing users to exert a certain degree on control over the generated content. This survey goes over the different techniques used for the automatic generation of terrains, which include different land formations such as mountains, valleys, rivers, shores, etc. These terrains have different uses such as simulation or entertainment which translates on different needs over the desired realism of the terrain and the degree of control that users have. Through time, different approaches have been proposed: repeating patterns that resemble those seen in nature; using software agents that imitate geological processes; using artificial intelligence techniques for pattern recognition and imitation of landscapes; or allowing users to interact with the system to draw desired terrain features. This review presents an overview of the area and discusses how different techniques adapt to the different needs and different stages of terrain creation.

Keywords: Automatic · Virtual · Landscape · Simulation · Videogames

1 Introduction

Procedural generation started as a solution for the constraints on memory availability when programming videogames: instead of saving all assets, just seeds and instructions would be saved. During the 80s, studies about the mathematical behavior of natural occurrences kicked-off several works, among them was the simulation of terrains, this is the inception of the area of Procedural Terrain Generation (PTG). Procedural generation refers to assets being automatically produced by a computer system, or with limited human input. The goal is to save time and resources through automatization [1].

The aim of this review is to provide an updated general overview of the procedural terrain generation area; without being too specific as in [2] where only evolutionary algorithms are presented or [3] which is focused on noise based methods; or too ample like in [4] and [5] where full videogames or virtual worlds are considered.

J. A. Carrasco-Ochoa et al. (Eds.): MCPR 2019, LNCS 11524, pp. 58–67, 2019.
https://doi.org/10.1007/978-3-030-21077-9_6

As noted in [2] the evaluation of the terrains is difficult. Execution-time comparisons are not always reported and depend not only on hardware, but on the size of the generated terrain and level of detail. Also, authors use different rendering algorithms, some do not use textures and only present the geometrical features of the terrains. Therefore, comparisons over quality cannot be objective. Hence, instead of a direct comparison, a taxonomy focusing on the generative capabilities of the algorithms is proposed.

In Sect. 2 the uses of the terrains, the generation methods and the taxonomy will be presented. In Sect. 3 there will be a discussion about the uses the methods. Finally, in Sect. 4 are the conclusions.

2 Terrain Classification

2.1 Uses of Terrains

Virtual terrains are used for videogames, simulations, movies and animations [6]. For simulations, real-world accuracy is necessary in terms of positions and spatial relations between features. If the simulation will be used for training, such as in self-driving cars, the level of detail needs to be as high as possible. In the case of movies only a single angle needs to be created, but it needs to look as realistic as possible. For animation, the above is true, but instead of realism, the terrain should be able to adapt to a desired aesthetic. In videogames the traversability is a desirable feature, this is that the players are able to traverse the terrain [7]. This is true for most games as there should be at least one route that the player can follow.

2.2 Generation Methods

There are four different families of generation techniques, they are the stochastic methods, the simulation methods, leaning methods and sketch-based methods. As shown in Table 1 each family comprises several methods.

Stochastic Methods. These methods use construction rules and parameters that try to imitate natural randomness, these rules could be applied recursively to increase detail. Control is exerted through these parameters and rules. Although these methods are very fast, they lack enough control and features are generated at random positions.

Fractal Methods. Fractals are geometric figures where each one of its parts has the same statistical character as the whole. These are created by repeating an instruction over an initial figure. The work of Prusinkiewicz [8] is one of the earliest fractal methods that could produce more than one type of landscape feature, it uses the midpoint displacement algorithm for generating mountains and squig curves for rivers. Chaotic fractals or noise can be used to create heightmaps that resemble natural terrains. Approximations of Perlin noise and fractional Brownian motion (fBm) functions are used in [9], and in [10] two approximations of pink noise are proposed.

Grammar-Based Methods. Grammars are rules for creation of formal languages in which growing sequences are applied to an initial state, in the work of Márak [11] these rules are used for deforming terrains and thus imitating the actions of erosion.

Table 1. Techniques for terrain generation.

Families	Stochastic	Simulation	Learning	Sketches
Techniques	Fractals	Geomorphological	Example-based	2D sketch
	Grammars	Ecosystem	Search-based	3D sketch
	Tiling	Agent-based	Inverse procedural generation	
	Parametrization			

Tiling. Tiling is a technique that improves terrains by dividing them into smaller areas. One of them is the Voronoi diagrams which divide the surface in cell-like structures. A different set of parameters can be used at each cell, in this way the monotony of noise algorithms is broken and transition between areas look more natural. Tiling is only able to make subdivision, for this reason it is used in combination with other methods [10].

Parametrization Methods. These techniques revolve around having building instructions which are controlled by the user, who adjusts their parameters. The variability of the results is limited by the number of controllable parameters. In [12] a method based on parametrization of tiles is presented. The authors present a terrain generator with latitude and dispersion parameters. The former controls the type of generated biomes and the later the size of the transition area between two biomes. Yin [13] proposes a method for creating mountain ridges, parameters are altitude, slope and influence area.

Simulation Methods. These methods try to emulate natural occurrences that generate terrains. There are three approaches: geomorphological simulations, ecosystem simulations, and agent-based simulations.

Geomorphological Simulation. Erosion is one of the main drivers of terrain formation and these approaches focus on its simulation for creating realistic and natural shapes. These methods are posterior to fractal ones [14]. In [15] the authors propose to use a layered representation which has information on the type of terrain. Upper layers provide material to lower ones, this simulates erosion of flowing water, and this is repeated until a slope threshold is reached. Erosion creates cliffs and canyons, while deposition generates plains. In the case of [16] tectonic uplift is also taken into consideration. This process elevates the whole terrain over time and thus can create higher mountain ranges.

Ecosystem Simulation. Two terrains with the same geomorphological formations can look very different by introducing vegetation and climate. Ecosystem simulation methods alter existing terrains. A layered model that includes vegetation is presented in [17]. There are four types of layers: bedrock, granular material, vegetation and dead vegetation. Elevation is determined by the sum of the first two layers. Vegetation is controlled by moisture and sun exposure. There are trees and shrubs, which have

density, height and age; and grass which only has density. In [18] the focus is over the accuracy of vegetation distribution. It needs two types of data: a heightmap and a biomass map. The later shows were vegetations is present and the former is the base terrain. Plants have the following data: species, spacing, relative size, closeness and coverage.

Agent-Based Simulation. These methods are considered separately as the agents do not base their actions on geomorphological events, usually they are divided into constructive and destructive. Doran and Parberry [19] define five types of agents: coastline agents which create initial landmass; smoothing agents that create plains; beach agents that flatten coastal areas; mountain generation agents; and river creation agents. Creation and smoothening are done randomly. Flattening and the creation of mountains are random but also include rules for certain control. The river agents connect a point in mountain areas with another in the coast following the lowest uphill gradient.

Learning Methods. Learning methods use real-world data, mostly in the form of images and digital elevation models (DEM) and try to imitate how real terrains look.

Example Methods. Terrain features are extracted from a set of examples. Zhou [20] proposes a method where small patches of mountains and canyons, extracted from real-world heightmaps, are copied and pasted over a sketch. Then, splines are used to create continuity between patches. In [21] extracted features and generated ones are combined to form full terrains. The construction of terrains is represented as a tree, where the leaves generated terrain parts and nodes are combination operators. When going further on the three, the level of detail increases, and these parts come from real-world images. In [22] Argudo creates a dictionary of atoms. The dictionary includes information on elevation, vegetation, light and water flow. To obtain an atom the terrain is divided into patches which are decomposed using sparse representation. When decomposing a new patch, if no combination of the existing atoms can recreate it a new atom is added to the dictionary. New terrains are created by combining the atoms.

Search-Based Methods. Most works are based on evolutionary algorithms (EAs), which rely on a set of examples that are evaluated using a fitness function. There are open problems with EAs: codification of terrains into chromosomes, fitness function design and improving variation of terrains. In the work of Frade [7] terrains are automatically evaluated by accessibility and obstacle length, but resulting terrains are too plain in comparison to real ones. On the other hand, Walsh [23] proposes a method where the user selects the parents for recombination, in this case the problem is the user fatigue.

Inverse Procedural Generation. This method recovers the idea of parameters, in here the system learns the parameters from examples. Emilien [24] proposes learning the distribution of objects such as trees. The number of objects in a radius will be reflected in a density function and stored in a brush, when this brush is used in another part of the map it will create a distribution of trees with the same density.

Sketch Methods. These methods improve the control over the generated terrains by giving the user the ability to place terrain features with painting-like interfaces. The disadvantage is that there is less realism in comparison with other methods.

2D Sketch. The method proposed in [13] the user draws a sketch map. Plains, mountains, canyons, rivers and lakes have color codes, and a skeleton is built upon a standard height that depends on the feature type. In [25] prevents the user draws a map that represents the altitude of areas, the system will then select adequate features. These features have areas of influence defined through Voronoi diagrams. Other works like [26] use sketches in the Y-axis, this allows to make changes in the terrain from a camera perspective that allows to see the horizon. The user draws silhouettes of the mountain features they want to make visible from that camera angle.

3D Sketch. Instead of making 2D strokes that will be later transformed into 3D features, some authors propose working with vectors that are 3D from the beginning. In the case of [27] curves are used. Once these curves are placed, the system will proceed to the voxelization, which is transforming the vectors into discrete approximations. Emilien [28] uses these 3D vectors to allow users to create river networks. Angle, position and length is decided by the user. If angles are too steep waterfalls are generated. The type of river depends on water flow and slope.

2.3 Taxonomy

A direct comparison between generation methods is difficult, generation speed and quality of generated terrains are not always reported or are subjective. This usually leads to comparisons that do not tell the potential uses of the generative methods, for this reason, the proposed taxonomy focuses on the generative capabilities of the algorithms and their needed inputs. As presented in Table 2, the taxonomy is composed of 7 categories that have mutually exclusive options.

Needed Input. There are algorithms that need pre-existing terrains to work on them, they can erode an existing terrain [11, 14, 29], can add vegetation or ecological behaviors [17, 18, 24], or can add volumetric features to terrains that did not have them [27, 30]. When a previous terrain is not needed, the user may input a map [25, 31], or a sketch [13, 20, 21, 32]. There may be a database which is used to create terrains [22, 33, 34], or the algorithm may create a terrain from nothing more than parameters input by the user [8, 10, 12, 16, 35–37].

Terrain Representation. The most used representations in literature are the heightmaps, sometimes referred as heightfields or digital elevation maps, these are grayscales images where the white color represent the highest altitude while the black is the lowest. This representation is compact but has the inability to show volumetric features, such as caves or overhangs. It also represents relative altitude relations and not absolute heights. In the opposite way, there are methods that represent the terrain differently, like multilayers that allow to represent materials [15, 29]; vegetation and water presence [22]; graphs [34]; volumetric pixels, or voxels, [30] and 3D curves [27].

Table 2. Taxonomy of terrain generation methods

Category	Options		
Needed input	Pre-existing terrain [11, 14, 17, 18, 24, 27, 29, 30]	No terrain needed [8, 10, 12, 13, 16, 20–22, 32–37]	
Terrain representation	Heightmap [8–14, 16–21, 23–26, 28, 30–40]	Other representation [15, 22, 27, 29, 34]	
Controllability	Random [8, 10, 14, 18, 29, 38]	Suggested control [7, 12, 17, 19, 21, 22, 31, 33, 37]	Precise control [13, 20, 24, 26–28, 32, 34, 39]
Terrain evaluation	User evaluated [8–40]	Automatic [7]	
Number of features	Single feature [24, 26, 29–31, 33]	Multiple features [16, 17, 19, 20, 25, 27, 28, 34, 40]	
Vegetation presence	Yes [12, 17, 18, 22, 24]	None/Texture only [7–11, 13–16, 19–21, 23, 25–40]	
Water presence	None/Flooding [7, 9–15, 17, 18, 22–27, 29–31, 33, 35–39]	Rivers [8, 16, 19–21, 32, 34, 40]	Others [19, 28]

Controllability. Earlier works, and those that are focused on a fast generation do not allow the user to control the position where the terrain features would be placed, positioning is random. These methods may be used to generate base terrains that would be used as input for other methods [8, 14, 29, 38], or when the terrains would be used as backgrounds and further editing is not needed [10, 18]. Suggested control is when the methods allow a certain degree of control over the positioning of the features. This may be achieved by subdividing the terrain in tiles and having inputs for each individual tile [12], by selecting examples for evolutive algorithms [7, 31]; by giving the system examples from real-world terrains [21, 22, 33]; by adjusting agents [19]; simulation parameters [12, 17]; or by adding constraints [37]. Finally, methods that provide a precise control over the position of the generated features are those that use sketches [13, 20, 26, 32, 39]; that allow for a three dimensional positioning of the features [27, 28, 30]; or that use brushes [24].

Terrain Evaluation. There are few proposals that aim to do an automatic evaluation of the terrains such as [7] where the terrains are evaluated according the distribution of ledges and the connectivity of the plain spaces. This is still a challenge in the area, as evaluation is done by users and is subjective.

Number of Features. There are algorithms that only allow to generate a single terrain feature, being the most common the generation of mountains [26, 29, 31, 33], others generate features such as caves [30] or vegetation that covers terrains [24]. It is noted that the mountain algorithms often flood areas below an altitude threshold, but these methods are not considered to be specialized in the generation of water bodies. On the other hand there are methods that are able to generate multiple types of terrain features, combining mountains with rivers [16, 34, 40], canyons [20], arches and overhangs [27],

vegetation [17] and plateaus [25]. There is also a specialization in the multiple features that a river network has including waterfalls [28]. And finally those that generate many features such as mountains, plains, plateaus, rivers and coastal plains [19].

Vegetation Presence. The presence of vegetation is important, because terrains would look very different depending on the plant types or if there is no vegetation at all. Many methods only include green textures that only work when looking at the terrains from afar, but when making a close-up they rapidly lose realism. Vegetation can be generated over an existing terrain [18, 24], or along with the terrain generation [12, 17, 22].

Water Presence. Other important feature are water bodies, these include rivers, lakes and shores. It is noticeable that most works that generate water bodies only generate rivers [8, 16, 19, 32, 34, 40] and canyons generated by those rivers [20, 21]. Those that generate other types of water bodies are not as common, in [28] waterfalls are generated, and in [19] shorelines are created using agents. Many works only flood lower areas, but there are no algorithms for specifically creating water bodies.

3 Discussion and Evaluation

Terrain generation methods have evolved in many ways during the last 30 years. Stochastic methods are the oldest ones and they produce terrains with basic features which are placed randomly. They are used for quickly generating base terrains that are altered using other methods or when trying to achieve real-time generation. By using GPUs, it is possible to create very big terrains during execution time with these methods.

When terrains need high geomorphological realism the simulation methods are the best ones, even if they are slow, the generated terrains are accurate. The most simulated event is erosion, specially water erosion. For this reason, the simulation of rivers and is common in the literature. Methods that use patches from real-world heightmaps and evolutive methods have proven to be close in terms of realism, but this depends on the used databases, which need to be big for improving results.

For videogames, the placement of assets or the use of pre-made maps is important, sketch methods perform the best to achieve this, as they allow to control the placing of the terrain features. Also, methods that can generate realistic water bodies and vegetation are preferred when using them for videogames because the player will traverse the terrain. The exception is flight simulators where the stochastic methods are enough because the terrain only serves as background.

When generating a terrain, the heightmap is the most common form of representation because of its low dimensionality which uses little memory, although it lacks the capability of representing hollow formations they can be later added with other algorithms.

Nowadays a highly competitive generator is one that would allow for high precision in the position of generated features; that allows to create several terrain features and that is capable to generate vegetation and water bodies. Automatic evaluation of terrains would be ideal as this would save more time to users, but this is still a problem that needs more attention because the proposals are limited to specific uses of the terrain.

4 Conclusion

In this paper the generation techniques and uses of the procedurally generated terrains were presented, and a taxonomy based on generative capabilities was proposed.

There are still many open research challenges in the area, such as the automatization of the evaluation methods, which has only been applied as a fitness function for evolutive methods. This is part of a bigger issue in the area: there is no standardized way of reporting results. Important data such as execution times or size of terrains is often missing. Also, the chosen render makes terrains look with different quality and geometrical characteristics are thus not properly evaluated, as a result, evaluations are subjective. Finding a way of making an objective comparison between generation methods is of interest in the area.

There are features that research has neglected such as coastal environments, cliffs, lakes, river deltas, wetlands, swamps, glaciers and glacial eroded terrains such as fjords. As research has advanced, the focus is now on adding more detail to the terrain through the simulation of ecological and weather processes, in the same way vegetation and water became important features of the terrains instead of just add-ons, it is possible that future research would require these simulations as an integral part of terrains.

References

1. Smith, G.: An analog history of procedural content generation. Presented at the Foundations on Digital Games (2015)
2. Raffe, W.L., Zambetta, F., Li, X.: A survey of procedural terrain generation techniques using evolutionary algorithms. In: 2012 IEEE Congress on Evolutionary Computation, pp. 1–8 (2012)
3. Rose, T.J., Bakaoukas, A.G.: Algorithms and approaches for procedural terrain generation - a brief review of current techniques. In: 2016 8th International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES), pp. 1–2 (2016)
4. Hendrikx, M., Meijer, S., Van Der Velden, J., Iosup, A.: Procedural content generation for games: a survey. ACM Trans Multimed. Comput. Commun. Appl. **9**, 1:1–1:22 (2013)
5. Smelik, R.M., Tutenel, T., Bidarra, R., Benes, B.: A survey on procedural modelling for virtual worlds. Comput. Graph. Forum **33**, 31–50 (2014). https://doi.org/10.1111/cgf.12276
6. Smelik, R.M., Tutenel, T., de Kraker, K.J., Bidarra, R.: A declarative approach to procedural modeling of virtual worlds. Computational Graphics **35**, 352–363 (2011)
7. Frade, M., de Vega, F.F., Cotta, C.: Automatic evolution of programs for procedural generation of terrains for video games. Soft Computing **16**, 1893–1914 (2012)
8. Prusinkiewicz, P., Hammel, M.: A fractal model of mountains and rivers. In: Graphics Interface, pp. 174–180. Canadian Information Processing Society (1993)
9. Helsing, J.K., Elster, A.C.: Real-time editing of procedural terrains (2015)
10. Olsen, J.: Realtime procedural terrain generation (2004)
11. Marák, I., Benes, B., Slavik, P.: Terrain erosion model based on rewriting of matrices. In: WSCG 1997, Winter School of Computer Graphics and Visualisation, pp. 341–350 (1997)
12. Choros, K., Topolski, J.: Parameterized and dynamic generation of an infinite virtual terrain with various biomes using extended voronoi diagram. J UCS. **22**, 836–855 (2016)

13. Yin, H.F., Zheng, C.W.: A practical terrain generation method using sketch map and simple parameters. IEICE Trans. Inf. Syst. **96**, 1836–1844 (2013)
14. Roudier, P., Peroche, B., Perrin, M.: Landscapes synthesis achieved through erosion and deposition process simulation. Comput. Graph. Forum **12**, 375–383 (1993)
15. Št'ava, O., Beneš, B., Brisbin, M., Křivánek, J.: Interactive terrain modeling using hydraulic erosion. In: Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 201–210. Eurographics Association (2008)
16. Cordonnier, G., et al.: Large scale terrain generation from tectonic uplift and fluvial erosion. Comput. Graph. Forum **35**, 165–175 (2016)
17. Cordonnier, G., et al.: Authoring landscapes by combining ecosystem and terrain erosion simulation. ACM Trans. Graph. TOG **36**, 134 (2017)
18. Onrust, B., Bidarra, R., Rooseboom, R., van de Koppel, J.: Ecologically sound procedural generation of natural environments. Int. J. Comput. Games Technol. **2017**, 17 (2017)
19. Doran, J., Parberry, I.: Controlled procedural terrain generation using software agents. IEEE Transactions on Computational Intelligence in AI and Games **2**, 111–119 (2010)
20. Zhou, H., Sun, J., Turk, G., Rehg, J.M.: Terrain synthesis from digital elevation models. IEEE Trans. Vis. Comput. Graph. **13**, 834–848 (2007)
21. Génevaux, J.-D., et al.: Terrain modelling from feature primitives. Comput. Graph. Forum **34**, 198–210 (2015)
22. Argudo, O., et al.: Coherent multi-layer landscape synthesis. Vis. Comput. **33**, 1005–1015 (2017)
23. Walsh, P., Gade, P.: Terrain generation using an interactive genetic algorithm. In: 2010 IEEE Congress on Evolutionary Computation (CEC), pp. 1–7. IEEE (2010)
24. Emilien, A., et al.: WorldBrush: interactive example-based synthesis of procedural virtual worlds. ACM Trans. Graph. TOG **34**, 106 (2015)
25. Mangra, A.P., Sabou, A., Gorgan, D.: TSCH algorithm-terrain synthesis from crude heightmaps. Rom. J. Hum.-Comput. Interact. **9**, 119 (2016)
26. Tasse, F.P., Emilien, A., Cani, M.-P., Hahmann, S., Dodgson, N.: Feature-based terrain editing from complex sketches. Computational Graphics **45**, 101–115 (2014)
27. Becher, M., Krone, M., Reina, G., Ertl, T.: Feature-based volumetric terrain generation. In: Proceedings of the 21st ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, p. 10. ACM (2017)
28. Emilien, A., et al.: Interactive procedural modelling of coherent waterfall scenes. Comput. Graph. Forum **34**, 22–35 (2015)
29. Beneš, B., Forsbach, R.: Parallel implementation of terrain erosion applied to the surface of mars. In: Proceedings of the 1st International Conference on Computer Graphics, Virtual Reality and Visualisation, pp. 53–57. ACM, New York (2001)
30. Dey, R., Doig, J.G., Gatzidis, C.: Procedural feature generation for volumetric terrains using voxel grammars. Entertain. Comput. **27**, 128–136 (2018)
31. Antoniuk, I., Rokita, P.: Procedural generation of adjustable terrain for application in computer games using 2D maps. In: Kryszkiewicz, M., Bandyopadhyay, S., Rybinski, H., Pal, S.K. (eds.) PReMI 2015. LNCS, vol. 9124, pp. 75–84. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19941-2_8
32. Gain, J., et al.: Terrain sketching. In: Proceedings of the 2009 Symposium on Interactive 3D Graphics and Games, pp. 31–38. ACM, New York (2009)
33. Cruz, L., et al.: Patch-based terrain synthesis. In: International Conference on Computer Graphics Theory and Applications, Berlin, France (2015)
34. Génevaux, J.-D., Galin, É., Guérin, E., Peytavie, A., Benes, B.: Terrain generation using procedural models based on hydrology. ACM Trans. Graph. TOG **32**, 143 (2013)