

Mountains Generation

Anna Boronina

I. INTRODUCTION

TERRAIN and landscape generation is one of the most important attributes of a game. Pyarelal et al. [1] claim that terrains appear in every three-dimensional game. They describe several ways to generate one: random terrain with interpolation or gradients used for smoothing, fractal terrain, agent-based, and search-based. Another proposition regarding terrain generation was made in 2005 by Ashlock et al. [2]. The idea is to use such a powerful concept as the L-System for map generation. Basically, they started from just one tile and then kept splitting into several new ones according to some rules. This project is based on Ashlock's paper.

II. METHODOLOGY

To run the algorithm, the following parameters should be set:

- 1) **STARTING HEIGHT** - the starting level for tile generation; it doesn't affect the visualization.
- 2) **SIZE** - size of one tile; must be a degree of 2.
- 3) **MOUNTAIN TYPE** - type of one mountain to generate: "calm", "hill", "steep".
- 4) **MOUNTAIN COMBINATION** - a two-dimensional array containing a combination of mountain tile types to construct. When constructed, all the tiles are connected.

During the generation, one of the last two options will be used - either one tile or a combination of different tiles.

A. Tiles creation

Below I discuss the creation of the tiles which is based on probabilities. Every type has five different tiles ranging from the highest to the lowest, and the probabilities table defining weights for going from one value to another.

For every type, the starting tile is A.

1) *Type "Calm"*: To generate an almost flat mountain, the algorithm uses values and probabilities from Fig. 1. The values are not high, as flat mountains should not have sudden jumps. For any value, the highest weights are given to value C which represents +0 on Z axis.

Tiles A and D are almost impossible to appear. The algorithm starts from tile A and goes to the only possible tile - C. Then, from C, the highest probability is to get other Cs.

2) *Type "Hill"*: To generate an average (not too flat but still walkable) mountain, the algorithm uses values and probabilities from Fig. 2. The values are not high, even smaller than for type "calm". Another difference is the probabilities table which gives higher weights to tiles A. Tiles A, B, D, E are more likely to appear for this type of mountain. Nevertheless, for any tile, the highest weights are still given to tile C which represents +0 on Z axis.

The starting tile is A, from which the mountain can develop in any direction with low probabilities for A or D.

3) *Type "Steep"*: To generate a steep (in other words, with a lot of jumps and peaks) mountain, the algorithm uses values and probabilities from Fig. 3. The values are higher than for previous types. Another difference is the probabilities table which gives higher weights to tiles A. Tiles C, D, and E are rare since steep mountains should not be too flat or go down.

(a)

A	B	C	D	E
2	1	0	-1	-2

(b)

	A	B	C	D	E
A	0.0	0.0	0.1	0.0	0.0
B	0.0	0.05	0.1	0.05	0.0
C	0.1	0.9	0.8	0.8	0.1
D	0.0	0.05	0.1	0.05	0.0
E	0.0	0.0	0.1	0.0	0.0

Fig. 1. Type "Calm": (a) Values (b) Probabilities table

(a)	A	B	C	D	E
	3	1.5	0	-1	-2

(b)		A	B	C	D	E
	A	0.1	0.075	0.3	0.2	0.2
	B	0.275	0.2	0.1	0.3	0.4
	C	0.35	0.45	0.3	0.3	0.35
	D	0.275	0.2	0.1	0.1	0.05
	E	0.1	0.075	0.2	0.1	0.0

Fig. 2. Type "Hill": (a) Values (b) Probabilities table

(a)	A	B	C	D	E
	9	7	0	-2.75	-4

(b)		A	B	C	D	E
	A	0.6	0.6	0.2	0.5	0.6
	B	0.15	0.05	0.2	0.15	0.1
	C	0.05	0.1	0.05	0.1	0.0
	D	0.1	0.05	0.03	0.15	0.15
	E	0.1	0.2	0.05	0.1	0.05

Fig. 3. Type "Steep": (a) Values (b) Probabilities table

B. Representation

The process of map generation can be seen as a tree where the root is the initial tiles, and the leaves are the final terrain. The next step is to parse this tree into a 2D map. Fig. 4 visualizes the process.

III. TAXONOMY

A. Online Versus Offline

Can be both.

Online: Since there are three types of tiles and two of them ("calm" and "hill") are walkable, the map can be created on the fly. Another reason to generate the map online is production time. According to Fig. 2 a map of 49 16x16 tiles takes 5 seconds to create. It will take significantly more time to generate a large number of tiles of size 32x32 and bigger due to the tree-like structure of a tile. Online generation of small tiles is an option for roguelike games that generate random levels online.

Offline: It is possible to generate all the necessary maps before gaming time, especially if big ones are required.

B. Necessary Versus Optional Content

Necessary. A landscape or terrain is probably the most important part of a level.

C. Random Seeds Versus Parametric Vectors

Both.

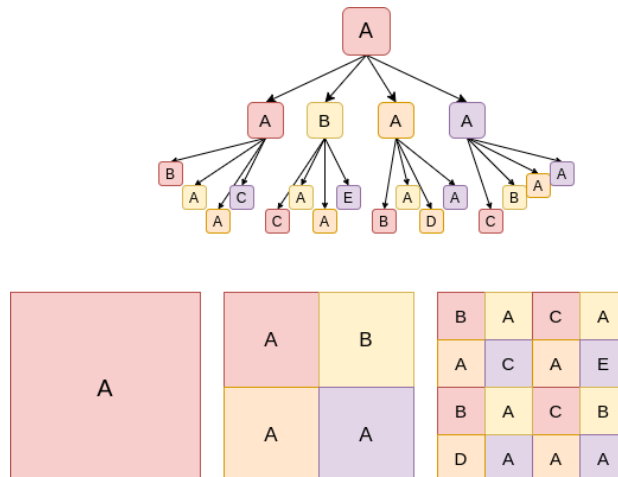


Fig. 4. Mapping a tree to a map

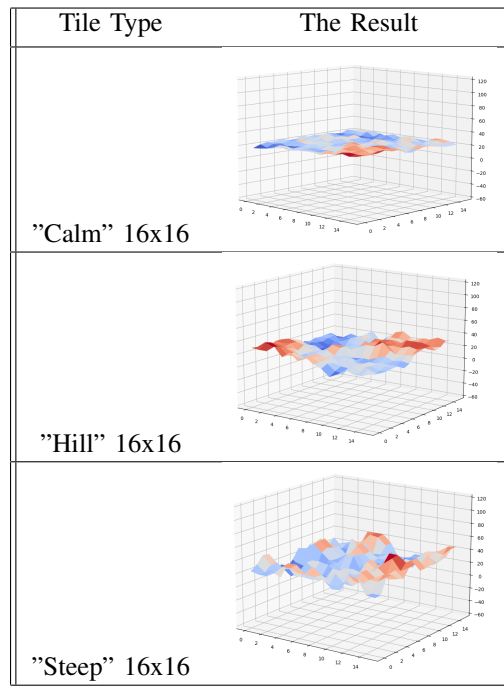


Fig. 5. Results: tile type and result

Parametric Vectors. The tables for each mountain type are predefined, as well as a user's input.

Random Seeds. Tables contain probabilities of a tile being followed by the same or another tile. When one tile is split into 4 next ones, they are chosen with weighted randomness.

D. Stochastic Versus Deterministic Generation

Stochastic. The algorithm utilizes a random choice of tiles.

E. Constructive Versus Generate-and-Test

Constructive. As mentioned before, at least two of the tiles are walkable. Therefore, there is no need to test the map if there is a path constructed of "hill" and "calm" tiles.

IV. RESULTS

The visual representation of generated tiles is presented in Fig. 5 and Fig. 6.

To evaluate the performance, I have used the following metrics:

- 1) **Generation time.** It should not be higher than 0.5 seconds for generation of 1 tile of size less than 16x16. Other sizes have shown to create rather noisy output. Connected tiles should not take longer than 5 seconds.
- 2) **Connectivity.** When generating a combination of tiles, they should be connected.

From Fig.5, the only map which took more than 5 seconds was 7x7 map of 16x16 tiles. As for connectivity, maps have transitions between the tiles.

The source code is in [this GitHub repository](#).

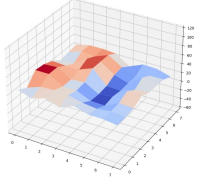
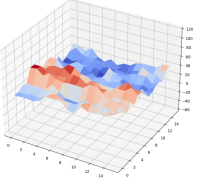
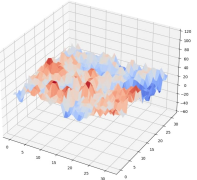
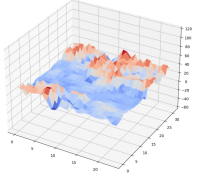
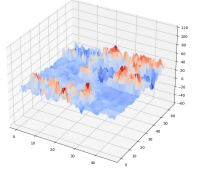
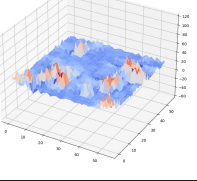
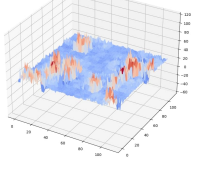
(Map Size)X(Tile Size)	Generation Time (s)	The Result
1x1 x 8x8	0.00892	
1x1 x 16x16	0.11135	
1x1 x 32x32	1.84991	
4x4 x 8x8	0.09771	
4x4 x 16x16	1.30003	
7x7 x 8x8	0.39618	
7x7 x 16x16	5.30813	

Fig. 6. Results: size and time

REFERENCES

- [1] Noor Shaker, Julian Togelius, and Mark J. Nelson. *Fractals, noise and agents with applications to landscapes*, pages 57–72. Springer International Publishing, Cham, 2016.
- [2] Daniel Ashlock, Stephen Gent, and Kenneth Bryden. Evolution of l-systems for compact virtual landscape generation. pages 2760–2767, 01 2005.