UDC 004.021

Sukhobrus Oleksandr, Pavlo Rehida.

REVIEW OF METODS OF ENVIRONMENT GENERATION IN COMPUTER GAMES

The article discusses different approaches to generating an environment in computer games. The article is divided into an overview of the main technology for creating natural landscapes and several modifications of methods of creating man-made structures made of rooms connected by corridors.

Target setting. Software generation has many advantages and disadvantages and for its proper use requires an understanding of its advantages and disadvantages.

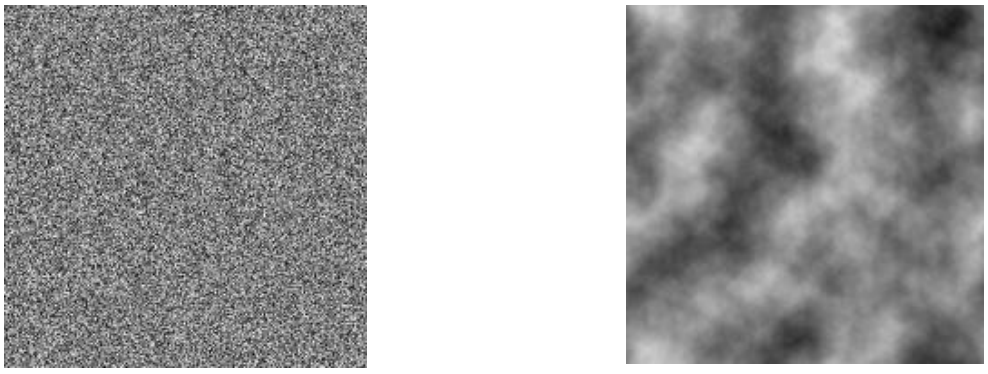
The research objective. The purpose of this article is to review the methods of generating the environment and highlight their advantages and disadvantages to understand the possibility of their use.

The task of environment generation in games can be divided into the following subtasks:

- Landscape generation
- Generation of structures

Landscape generation. The generated landscape serves as a background for the main game activities and needs to be supplemented by other methods of generation, or pre-created objects, which is its main drawback.

The main way to generate a landscape is Perlin noise [2], or its modification. It was created by Ken Perlin in 1983 to generate images, but was later used in computer games to generate landscapes. Compared with the noise in which all values are independent, Perlin noise is smooth transitions between adjacent values, as shown in Fig. 1.



a) b)

Fig. 1. The result is obtained by:
a) noise with independent values; b) Perlin noise

The differences in obtaining Perlin noise from completely random noise is to assign to each point of the grid with a certain step a unit vector (gradient vectors), rather than a random number to each point in space, as shown in Fig. 2.

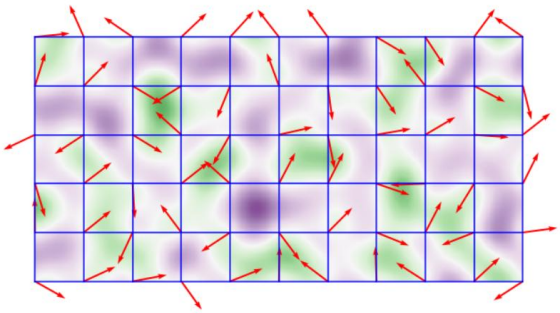


Fig .2. Perdlin two-dimensional noise grid with gradient vectors

In the process of obtaining the generation result, the vectors are used to further calculate the noise values at points lying in the cells of the grid. To calculate the values at each point, the operations of the scalar product of gradient vectors on the corresponding vectors of distances to the point are performed. The result of this process is 4 values for two-dimensional noise and 8 for four-dimensional, which determines the complexity of the algorithm $O(2^n)$. The final step in determining the values at a point is to interpolate the values obtained in the previous step.

Generation of structures. Thanks to the ability to create incentives for the player, structures can act as independent objects of the game environment or complement the generated game landscapes. Generation of structures is usually performed by placing certain pre-created by the developer or generated in the process parts in a certain order provided by the game logic. In most cases, such parts are the rooms and corridors that connect them.

Static grid. One of the first and easiest methods of generation used today is to place rooms in the middle of the cells of a static grid and then connect them with the help of corridors. Due to the ability to easily predict the number and location of rooms, this method is well suited for the use of pre-created rooms. The downside is the obvious layout of the rooms. An example of a game using this method is shown in Fig. 3.

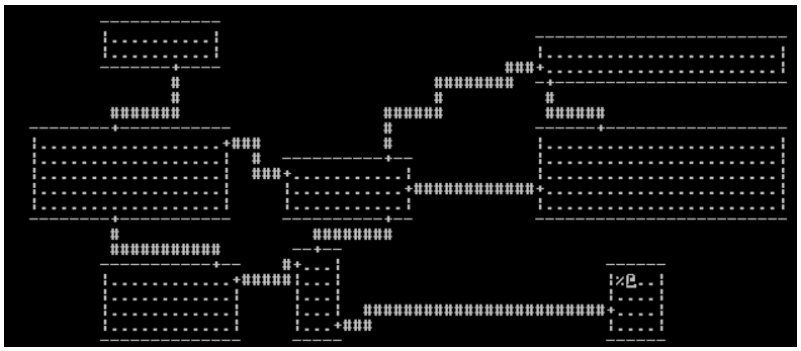


Fig. 3 Level generated using a static grid

Binary space partitioning [3]. This method belongs to the more complex methods of placing rooms. By dividing the space of the playing field into two parts, a binary tree is created, the child of each node are areas that have adjacent edges.

This method allows you to add more randomness to the placement of rooms on the playing field, but at the same time increases the complexity of the placement of corridors and rooms created by the developer. An example of the result of this method is shown in Fig. 4.

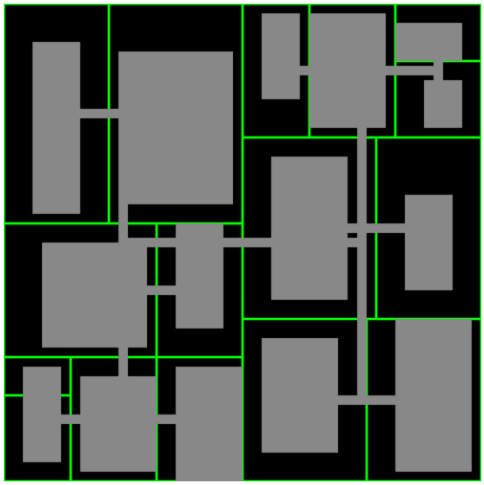


Fig. 4 Structure generated using a binary tree

Tunneling method [4]. The method is based on the gradual laying of corridors on the playing field and the creation of rooms next to them. The main disadvantage is the complexity of writing algorithms for tunneling and the direct relationship between their quality and the quality of the generated structure. The advantages include the fact that with this method you can achieve the generation of a structure that looks designed by hand. . An example of the result of this method is shown in Fig. 5

Fig. 5 Structure generated using the tunneling method

Conclusions. The article describes the advantages and disadvantages of methods of generating the environment. Such disadvantages include the need to supplement the generation of landscapes to provide incentives for the game, the difficulty of including manually created elements in structures generated by binary space division, and the predictability of the environment created by static grid.

References

1. Albert Carri´on D´ıaz. (Spring 2015). Procedural generation applied to a video (pp. 19-20)

2. Eevee. (2016, May 29). Perlin noise. [Blog post] Retrieved from https://eev.ee/blog/2016/05/29/perlin-noise/

3. Lluis Esquerda. (2013, December 22). Dungeon generation using BSP trees. [Blog post] Retrieved from https://eskerda.com/bsp-dungeon-generation/

4. Josh Ge. (2014, June 20). Mapgen: Tunneling Algorithm. [Blog post] Retrieved from https://www.gridsagegames.com/blog/2014/06/mapgen-tunneling-algorithm/

AUTHORS

Oleksandr Sukhobrus– student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: sanysuh@gmail.com

Pavlo Rehida (supervisor) – assistant professor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: pavel.regida@gmail.com