УДК 10.23.256 https://doi.org/10.54251/2616-6429.2026.02.n12

S.S. Beknazarova*, K.K. Beknazarova
Dsc, professor, Tashkent University of Information Technologies named after Muhammad Al- Khwarizmi, Tashkent, Uzbekistan
Researcher, Tashkent University of Information Technologies named after Muhammad Al- Khwarizmi, Tashkent, Uzbekistan
*Corresponding author: *saida.beknazarova@gmail.com*

PROCEDURAL TEXTURE SYNTHESIS IN 3D RECONSTRUCTION

Abstract

The article explores the advantages of procedural texture synthesis in 3D reconstruction of historical objects and creating high-quality textures in virtual environments. The main focus is on methods for creating unique and realistic textures using fractal and space curves, such as the Cantor set, Gilbert and Gosper curves. The core contribution of the work lies in the systematic application of fractal geometry and space-filling curves to texture generation. The methodology rigorously examines three fundamental mathematical constructs: the Cantor set for creating discontinuous and dust-like patterns, the Gilbert curve for locality-preserving 2D space traversal, and the Gosper curve for generating hexagonal and snowflake-like tiling structures. By parameterizing these curves and integrating them with fractal noise functions, the authors propose a hybrid algorithmic pipeline that synthesizes textures at runtime, allowing for infinite variation and seamless detail across different levels of zoom (detail-invariant synthesis).

Keywords:

procedural texture synthesis, Cantor set, Gilbert curve, Gosper curve, fractal technologies, 3D reconstruction, virtual environment, computer graphics

Introduction

In recent years, the increasing demand for digitizing historical objects and bringing them to life in virtual environments has brought computer graphics and visualization technologies to a new level. In particular, procedural texture synthesis, which automatically creates textures for surfaces based on mathematical rules, is helping to make 3D models of historical objects more realistic and creative. In this process, classic fractal structures and curves help to create realistic details[1].

There are two main methods for creating textures in the field of computer graphics: hand-made image textures and procedural textures based on mathematical rules. The first method is labor-intensive and often limited to repeating patterns. Procedural texture synthesis, on the other hand, allows for the automatic creation of complex patterns based on mathematical algorithms [2].

Elements of fractal geometry, including the Cantor set, Gilbert curve, and Gosper curve, are widely used to generate complex patterns that resemble natural structures. These curves, due to their self-repeating nature, create visual structures close to natural forms. Therefore, they are an effective tool for texture synthesis[3,4].

Also, fractal-based procedural textures can be generated in real time. This expands the possibilities of their use in 3D graphics, game engines, virtual museums and VR systems. In addition, during the reconstruction of historical objects, real texture data is often not enough. Using fractal algorithms, it is possible to artificially restore non-existent or damaged surfaces. As a result, procedural texture synthesis is becoming an important tool for preserving historical heritage, conducting scientific research and creating educational virtual environments [5].

3D reconstruction of historical objects is widely used in many fields, including archaeology, cultural heritage conservation, education and virtual tours. Traditional hand-created textures are time-

consuming and resource-intensive, and are limited to repetitive patterns. Therefore, procedural texture synthesis using mathematical structures such as Cantor sets, Gilbert and Gosper curves effectively creates high-quality, parametric, and visually complex patterns [6].

Today, digital reconstruction of historical objects is widely used in the fields of archaeology, cultural heritage preservation, and tourism. High-quality visual images are required for the creation of virtual museums, 3D tours, and digital archives. In this process, textures are one of the main elements that provide a realistic view of the object [7].

Traditional texture creation methods often require large image files and create the problem of repetitive patterns. Procedural texture synthesis, on the other hand, reduces memory consumption and provides flexible parametric control because it is based on mathematical formulas [8].

Using fractal structures, it is possible to create complex textures with a natural appearance. Fractal structures such as the Cantor set allow for surface segmentation, while Gosper and Gilbert curves allow for the generation of complex patterns. The methods also work effectively in real-time graphics systems. This allows them to be used in game engines, VR systems, and scientific visualization. Procedural textures also allow for the modeling of various historical materials - stone, brick, marble, or ancient wall patterns - by changing parameters. As a result, fractal-based texture synthesis is of significant scientific and practical importance in 3D reconstruction of historical objects [9,10].

This research aims to solve the following problems: identify texture generation methods based on fractal structures (Cantor set); increase visual subtlety and complexity using Gilbert and Gosper curves; evaluate the effectiveness of applying procedural textures to historical objects undergoing 3D reconstruction[1-3].

Literature review

In the field of computer graphics, much research has been done on procedural texture synthesis. Ebert et al. (2003) presented the theoretical foundations of procedural texture creation in their famous work “Texturing and Modeling: A Procedural Approach”. The authors showed methods for creating complex visual patterns using fractal algorithms, noise functions, and mathematical models. This work is considered one of the fundamental sources in the field of procedural texture synthesis[1].

Lagae et al. (2010) studied the application of procedural textures in real-time graphics systems. Their research showed that high performance can be achieved using graphics processing units (GPUs). This work laid the foundation for the widespread use of procedural textures in real-time rendering and game engines [2,3].

Akenine-Möller, Haines, and Hoffman (2019) analyzed the role and importance of texture synthesis in modern real-time rendering technologies. The authors have shown the role of fractal structures, space-filling curves, and shader technologies in graphics systems. These studies serve as an important theoretical basis for the development of texture generation algorithms based on fractal curves [4,5].

Methodology

The research methodology was developed by combining fractal geometry, procedural texture synthesis, and computer graphics algorithms. At the first stage, the mathematical properties of the Cantor set, Gilbert, and Gosper curves were studied and their application in texture generation was analyzed. The Cantor set was used as an iterative fractal algorithm to segment surfaces and create the basic structure of the texture structure. In this method, the initial line or the central part of the surface is divided according to certain rules, and new fractal segments are generated in each iteration. As a result, complex, self-repeating fractal patterns appear on the surface. This mathematical model acts as the main skeleton for the texture structure. In the second stage, the space-filling curves Gilbert and Gosper curves were used. These curves have the property of continuously covering two-dimensional space, allowing for a smooth and orderly arrangement of texture elements. Using the Gilbert curve algorithm, pixel coordinates were generated sequentially, covering the entire surface. The Gosper

curve was used to create complex fractal patterns and give a natural look to the texture layers. These algorithms were modeled based on the L-system and iteration rules. The resulting curves were used as a coordinate map when generating texture elements. In the third stage, the developed fractal algorithms were implemented in a graphics programming environment. Shader technologies were used to make the texture generation process work in real time, in particular, procedural textures were created using GLSL fragment shaders. During the testing process, the created textures were applied to various 3D models - walls of historical buildings, stone surfaces and architectural elements. The results were evaluated in terms of visual quality, level of complexity and computational efficiency. The results showed that fractal-based procedural textures require less memory than traditional bitmap textures and provide a high level of flexibility.

Many well-known fractals are considered close to the Cantor set, so the fractal properties of the Cantor set are of great importance. The construction of the Cantor set begins with the removal of the middle part from the unit cut, while the ends of the cut are not included. That is, the cut [0,1] is the initial set, and the first step is to delete the open interval (1/3, 2/3). In the next and all subsequent steps, the middle part is removed from all cuts (excluding the ends) of the current step. In this order, we obtain the Cantor set with a sequence size d ≈ 0.9542 [11,12].

The Cantor set of size d = 1. It is possible to construct a Cantor set of size d=1 by moving from a straight line to a plane. The next example belongs to Magdi Muhammad. The initial set consists of a single square and we assume that the vertices are (0,0), (1,0), (1,1) and (0,1). At each step we generate squares that are four times smaller than the previous square. The limit set of this construction is a self-similar set with N = 4 and a similarity coefficient of $r = \frac{1}{4}$. Its size

$$d = \frac{\log(4)}{\log(4)} = 1$$

As can be seen from the constructions, the resulting set is a Kantor set, compact, perfect, and completely continuous.

$$\omega_0(a,x,y) = \frac{a^2-x^2}{2a} \wedge_0 \frac{a^2-y^2}{2a} \geq 0. \quad (1)$$

d=1 Based on the condition of the dimensional Kontor set, that is, we construct an iterative process, and as a result we obtain the following [6-8].

$$\omega_n(a,x,y) = \omega_{n-1}\left(\frac{a}{4}, x-\frac{3a}{4}, y-\frac{a}{4}\right) \vee_0$$
$$\vee_0 \omega_{n-1}\left(\frac{a}{4}, x-\frac{a}{4}, y-\frac{3a}{4}\right) \vee_0 \omega_{n-1}\left(\frac{a}{4}, x+\frac{3a}{4}, y+\frac{a}{4}\right) \vee_0$$
$$\vee_0 \omega_{n-1}\left(\frac{a}{4}, x+\frac{a}{4}, y+\frac{3a}{4}\right) \vee_0 (x(a^2-x^2) = 0 \wedge_0 (a^2-y^2) \geq 0) \vee_0$$
$$\vee_0 (y(a^2-y^2) = 0 \wedge_0 (a^2-x^2) \geq 0),\ n = 1,2,3,\ldots \quad (2)$$

Gilbert curve ω_0- empty set.

ω_0 we get the following as:

$$(\omega_0(x,y) = (-1-x^2) \geq 0). \quad (3)$$

Now, to control the order of Gilbert curves, we need the following formulas: [7]:

$$\begin{cases} m_0 = 0 \\ m_n = 2m_{n-1} + a. \end{cases} \quad (4)$$

Here m_n - *n*- size of the ordered Gilbert line (*a*- unit of measure).

$$f_1(x,y) = \big((y=0) \wedge_0 (x-m_{n-1}) \wedge_0 (m_{n-1}+a-x)\big) \geq 0;$$

(lower connecting line)

$$f_2(x,y) = \big((x-m_{n-1}=0) \wedge_0 (y-m_{n-1}) \wedge_0 (m_{n-1}+a-y)\big) \geq 0;$$

(left connecting line)

$$f_3(x,y) = \big((y-2m_{n-1}-a=0) \wedge_0 (x-m_{n-1}) \wedge_0 (m_{n-1}+a-x)\big) \geq 0.$$

(top connecting line).

Based on the recursive algorithm, we get: $\omega_n(x,y) = \omega_{n-1}(x,y) \vee_0 f_1(x,y) \vee_0 \omega_{n-1}(m_{n-1} - y, x - m_{n-1} - a) \vee_0$

$$\vee_0 f_2(x,y) \vee_0 \omega_{n-1}(x, y - m_{n-1} - a) \vee_0 f_3(x,y) \vee_0$$
$$\vee_0 \omega_{n-1}(y - m_{n-1} - a, x - m_{n-1} - a), n = 1,2,3, \ldots \quad (5)$$

The Gosper curve is relatively similar to the Serpin curve, except that the angles of the Gosper curve are tilted relative to the axes OX and OY: $\omega_1(a,x,y) = \left(\frac{a^2}{4} - \left(x - \frac{a}{2}\right)^2\right) \wedge_0 (a_{11}^2 - y^2) \geq 0.$ (6)

Here a_{11}- a sufficiently small number (the thickness of the line).

Now we calculate the value of the angle of deviation in the axis system relative to the fixed coordinate system and apply the translation and rotation formulas here. $\varphi = \arctan\left(\frac{\sqrt{3}}{5}\right), a_{ky} = \frac{a}{\sqrt{7}}, a_{mv} = a_{ky}\frac{\sqrt{3}}{2}.$

$x_{ky} = x\cos(\varphi) + y\sin(\varphi)\,;\; y_{ky} = -x\sin(\varphi) + y\cos(\varphi) + a_{mv}.$ (7)

Now, using recursion, we get: $\omega_n(a,x,y) = \omega_{n-1}(a_{ky}, x_{ky}, y_{ky} - a_{mv}) \vee_0$

$$\vee_0 \omega_{n-1}(a_{ky}, \left(x_{ky} - \frac{3a_{ky}}{2}\right)\cos\left(\frac{2\pi}{3}\right) + y_{ky}\sin\left(\frac{2\pi}{3}\right),$$
$$-\left(x_{ky} - \frac{3a_{ky}}{2}\right)\sin\left(\frac{2\pi}{3}\right) + y_{ky}\cos\left(\frac{2\pi}{3}\right)) \vee_0 \omega_{n-1}\left(a_{ky}, x_{ky} - \frac{a_{ky}}{2}, y_{ky}\right) \vee_0$$
$$\vee_0 \omega_{n-1}(a_{ky}, (x_{ky} - \frac{a_{ky}}{2})\cos\left(-\frac{2\pi}{3}\right) + y_{ky}\sin\left(-\frac{2\pi}{3}\right),$$
$$-\left(x_{ky} - \frac{a_{ky}}{2}\right)\sin\left(-\frac{2\pi}{3}\right) + y_{ky}\cos\left(-\frac{2\pi}{3}\right)) \vee_0 \omega_{n-1}(a_{ky}, x_{ky}, y_{ky} + a_{mv}) \vee_0$$
$$\vee_0 \omega_{n-1}(a_{ky}, x_{ky} - a_{ky}, y_{ky} + a_{mv}) \vee_0$$
$$\vee_0 \omega_{n-1}(a_{ky}, \left(x_{ky} - \frac{5a_{ky}}{2}\right)\cos\left(-\frac{2\pi}{3}\right) + y_{ky}\sin\left(-\frac{2\pi}{3}\right),$$
$$-\left(x_{ky} - \frac{5a_{ky}}{2}\right)\sin\left(-\frac{2\pi}{3}\right) + y_{ky}\cos\left(-\frac{2\pi}{3}\right)), n = 2,3,4, \ldots \quad (8)$$

Fractals consisting of circles. Currently, there are several methods for constructing fractal equations: the method of L-systems, the method of iterative function systems, etc. The design environment of the R-function algebra logic method, which is different from them, makes it possible to construct fractal equations. Then, based on these equations, a visual representation of fractals can be constructed. So, below, the construction of fractal equations consisting of circles based on the RFM method of V.L. Rvachev is considered. The equation of the outer circle is defined as follows: $\omega_{00} = \omega_{00}(R,x,y) = (R^2 - x^2 - y^2 \geq 0),$

The equation of a circle bounded by a circle takes the following form: $\omega_0 = \omega_{00} \wedge_0 (x^2 + y^2 - (R-a)^2 \geq 0),$

here *a*- thickness of the circle (thickness of the circle *2a*), *R*- radius of the outer circle, $\alpha = \frac{2\pi}{k}$; *k* - number of inner loops after each iteration *k=2,3,4,...*

Here, using iteration, we get: $\omega_n(R,x,y) = \omega_0(R,x,y) \vee_0 \omega_{n-1}\left(\frac{R}{3}, x, y\right) \vee_0$

$$\vee_0 \omega_{n-1}\left(\frac{R}{3}, x - \frac{2R}{3}\cos(0), y - \frac{2R}{3}\sin(0)\right) \vee_0$$
$$\vee_0 \omega_{n-1}\left(\frac{R}{3}, x - \frac{2R}{3}\cos(\alpha), y - \frac{2R}{3}\sin(\alpha)\right) \vee_0$$

$$\vee_0 \omega_{n-1}\left(\frac{R}{3}, x-\frac{2R}{3}\cos(2\alpha), y-\frac{2R}{3}\sin(2\alpha)\right)\vee_0 \ldots \vee_0$$
$$\vee_0 \omega_{n-1}\left(\frac{R}{3}, x-\frac{2R}{3}\cos\big((k-1)\alpha\big), y-\frac{2R}{3}\sin((k-1)\alpha)\right)\geq 0; n=1,2,3,\ldots \quad (9)$$

Now let's look at the case of k=6. In this case, we get exclusive ring fractals.

It can be shown that if the inner circles do not collide with each other at k<6, the inner circles collide at k=6, and the inner circles intersect at k>6. Now let's consider the case where there are two circles inside a large circle. These circles are intersected. In turn, two more circles are formed in the inner circles, etc. We construct the equation of the same fractal [9]. In this case, the equation of the fractal in step 1 takes the following form $\omega(R,x,y)=(R^2-x^2-y^2\geq 0)\wedge_0 (x^2+y^2-(R-a)^2\geq 0)$. (10)

Trivial intersecting circle fractals. After applying the iteration procedure, we obtain the following $\omega_1(R,x,y)=\omega_0(R,x,y)\vee_0 \omega_0\left(\frac{R}{2},x,y-\frac{R}{2}\right)\vee_0 \omega_0\left(\frac{R}{2},x,y+\frac{R}{2}\right)\geq 0$,

$$\omega_2(R,x,y)=\omega_0(R,x,y)\vee_0 \omega_1\left(\frac{R}{2},x,y-\frac{R}{2}\right)\vee_0 \omega_1\left(\frac{R}{2},x,y+\frac{R}{2}\right)\geq 0,$$

…

$$\omega_n(R,x,y)=\omega_0(R,x,y)\vee_0 \omega_{n-1}\left(\frac{R}{2},x,y-\frac{R}{2}\right)\vee_0 \omega_{n-1}\left(\frac{R}{2},x,y+\frac{R}{2}\right)\geq 0; n=1,2,3 \quad (11)$$

Now we consider the case where the inner circles intersect and decrease. For this purpose, we introduce a decreasing coefficient λ. We determine the equation of the intersecting circles as in the first problem $\omega_0(R,x,y)=(R^2-x^2-y^2\geq 0)\wedge_0 (x^2+y^2-(R-a)^2\geq 0)$,

$$\alpha=\frac{2\pi}{k};$$

k - number of inner loops after each iteration *k=2,3,4,...*

l - the coefficient of reduction of the inner circles after each iteration, *l=2,3,4,...*

R - the radius of the outer circle.

Using the iteration procedure, we can obtain the following [6].

$$\omega_n(R,x,y)=\omega_0(R,x,y)\vee_0 \omega_{n-1}\left(\frac{R}{l},x,y\right)$$
$$\vee_0 \omega_{n-1}\left(\frac{R}{l},x-\frac{(l-1)R}{l}\cos(0),y-\frac{(l-1)R}{l}\sin(0)\right)\vee_0$$
$$\vee_0 \omega_{n-1}\left(\frac{R}{l},x-\frac{(l-1)R}{l}\cos(\alpha),y-\frac{(l-1)R}{l}\sin(\alpha)\right)\vee_0$$
$$\vee_0 \omega_{n-1}\left(\frac{R}{l},x-\frac{(l-1)R}{l}\cos(2\alpha),y-\frac{(l-1)R}{l}\sin(2\alpha)\right)\vee_0 \ldots \vee_0$$
$$\vee_0 \omega_{n-1}\left(\frac{R}{l},x-\frac{(l-1)R}{l}\cos((k-1)\alpha),y-\frac{(l-1)R}{l}\sin((k-1)\alpha)\right)\geq 0; n=1,2,3,\ldots \textit{(12)}$$

Let's look at constructing a tree equation from circles. The ends of the interval *(x1,y1)* va *(x2,y2)* be points. Given points *(x1,y1)* and *(x2, y2)* construct the equation of a straight line passing freely through [10].

$$f(x_1,y_1,x_2,y_2,x,y)=\left(\left(\frac{1}{2}(x_2-x_1)\right)\cos\left(\arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right)+\right.$$
$$+(y_2-y_1)\sin(\arctan(\frac{y_2-y_1}{x_2-x_1})))^2-((x-x_1)\cos\left(\arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right)+$$
$$+(y-y_1)\sin\left(\arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right)-(\frac{1}{2}(x_2-x_1)\cos\left(\arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right)+$$

$$+(y_2-y_1)\sin\left((\arctan\left(\frac{y_2-y_1}{x_2-x_1}\right))^2\geq 0\right))\wedge_0(a^2-$$

$$-\left(-(x-x_1)\sin\left(\arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right)\right)+(y-y_1)\cos\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right)^2\geq 0)$$

here *a* - height of the gap (height of the gap *2a*).

If *k* if it is even, then$\varphi_0=0$, else $\varphi_0=\frac{\alpha}{2}$.

n=1 we have the following equation :$\alpha=\frac{2\pi}{k}$

$$\omega_1(x,y)=f(0{,}0,R\cos(\varphi_0+0),R\sin(\varphi_0+0),x,y)\vee_0$$
$$\vee_0 f(0{,}0,R\cos(\varphi_0+\alpha),R\sin(\varphi_0+\alpha),x,y)\vee_0$$
$$\vee_0 f(0{,}0,R\cos(\varphi_0+2\alpha),R\sin(\varphi_0+2\alpha),x,y)\vee_0\ldots\vee_0$$
$$\vee_0 f(0{,}0,R\cos(\varphi_0+(k-1)\alpha),R\sin(\varphi_0+(k-1)\alpha),x,y)$$

n=2,3,4… da

$$\alpha=\frac{2\pi}{k^{n-1}};\ k_1=-\left[\frac{k}{2}\right];\ R_{n-1}=2R\left(1-\frac{1}{2^{n-1}}\right);R_n=2R\left(1-\frac{1}{2^n}\right);$$

R_n – *n*- radius of the circular boundaries in iteration *(R_1=R).*

if *k* is even, then$k_2=\left[\frac{k}{2}\right]$, aks holda $k_2=\left[\frac{k}{2}\right]-1$.

[x] - x the whole number.

Using the iteration procedure, we get: $\omega_{nx1}(x,y)=f(R_{n-1}\cos(\varphi_0+\alpha),R_{n-1}\sin(\varphi_0+\alpha),$

$$R_n\cos\left(\varphi_0+\alpha+\frac{(\varphi_0+k_1\alpha)}{k}\right),R_n\sin\left(\varphi_0+\alpha+\frac{(\varphi_0+k_1\alpha)}{k}\right),x,y)\vee_0$$
$$\vee_0 f(R_{n-1}\cos(\varphi_0+\alpha),R_{n-1}\sin(\varphi_0+\alpha),$$
$$R_n\cos\left(\varphi_0+\alpha+\frac{(\varphi_0+(k_1+1)\alpha)}{k}\right),R_n\sin\left(\varphi_0+\alpha+\frac{(\varphi_0+(k_1+1)\alpha)}{k}\right),x,y)\vee_0$$
$$\vee_0 f(R_{n-1}\cos(\varphi_0+\alpha),R_{n-1}\sin(\varphi_0+\alpha),$$
$$R_n\cos\left(\varphi_0+\alpha+\frac{(\varphi_0+(k_1+2)\alpha)}{k}\right),R_n\sin\left(\varphi_0+\alpha+\frac{(\varphi_0+(k_1+2)\alpha)}{k}\right),x,y)\vee_0\ldots\vee_0$$
$$\vee_0 f(R_{n-1}\cos(\varphi_0+\alpha),R_{n-1}\sin(\varphi_0+\alpha),$$
$$R_n\cos\left(\varphi_0+\alpha+\frac{(\varphi_0+(k_2)\alpha)}{k}\right),R_n\sin\left(\varphi_0+\alpha+\frac{(\varphi_0+(k_2)\alpha)}{k}\right),x,y)$$

$$\omega_{nx2}(x,y)=f(R_{n-1}\cos(\varphi_0+2\alpha),R_{n-1}\sin(\varphi_0+2\alpha),$$
$$R_n\cos\left(\varphi_0+2\alpha+\frac{(\varphi_0+k_1\alpha)}{k}\right),R_n\sin\left(\varphi_0+2\alpha+\frac{(\varphi_0+k_1\alpha)}{k}\right),x,y)\vee_0$$
$$\vee_0 f(R_{n-1}\cos(\varphi_0+2\alpha),R_{n-1}\sin(\varphi_0+2\alpha),R_n\cos\left(\varphi_0+2\alpha+\frac{(\varphi_0+(k_1+1)\alpha)}{k}\right),$$
$$R_n\sin\left(\varphi_0+2\alpha+\frac{(\varphi_0+(k_1+1)\alpha)}{k}\right),x,y)\vee_0$$
$$\vee_0 f(R_{n-1}\cos(\varphi_0+2\alpha),R_{n-1}\sin(\varphi_0+2\alpha),R_n\cos\left(\varphi_0+2\alpha+\frac{(\varphi_0+(k_1+2)\alpha)}{k}\right),$$
$$R_n\sin\left(\varphi_0+2\alpha+\frac{(\varphi_0+(k_1+2)\alpha)}{k}\right),x,y)\vee_0\ldots\vee_0$$
$$\vee_0 f(R_{n-1}\cos(\varphi_0+2\alpha),R_{n-1}\sin(\varphi_0+2\alpha),R_n\cos\left(\varphi_0+2\alpha+\frac{(\varphi_0+k_2\alpha)}{k}\right)\ R_n\sin\left(\varphi_0+2\alpha+\frac{(\varphi_0+k_2\alpha)}{k}\right),x,y)\ (13)$$

$1 \le i \le k^{n-1}$ for we have the following:

$$\omega_{nxi}(x,y) = f(R_{n-1}\cos(\varphi_0 + i\alpha), R_{n-1}\sin(\varphi_0 + i\alpha),$$
$$R_n\cos\left(\varphi_0 + i\alpha + \frac{(\varphi_0 + k_1\alpha)}{k}\right), R_n\sin\left(\varphi_0 + i\alpha + \frac{(\varphi_0 + k_1\alpha)}{k}\right), x, y) \vee_0$$
$$\vee_0 f(R_{n-1}\cos(\varphi_0 + i\alpha), R_{n-1}\sin(\varphi_0 + i\alpha), R_n\cos\left(\varphi_0 + i\alpha + \frac{(\varphi_0 + (k_1+1)\alpha)}{k}\right),$$
$$R_n\sin\left(\varphi_0 + i\alpha + \frac{(\varphi_0 + (k_1+1)\alpha)}{k}\right), x, y) \vee_0$$
$$\vee_0 f(R_{n-1}\cos(\varphi_0 + i\alpha), R_{n-1}\sin(\varphi_0 + i\alpha), R_n\cos\left(\varphi_0 + i\alpha + \frac{(\varphi_0 + (k_1+2)\alpha)}{k}\right),$$
$$R_n\sin\left(\varphi_0 + i\alpha + \frac{(\varphi_0 + (k_1+2)\alpha)}{k}\right), x, y) \vee_0 \ldots \vee_0$$
$$\vee_0 f(R_{n-1}\cos(\varphi_0 + i\alpha), R_{n-1}\sin(\varphi_0 + i\alpha), R_n\cos\left(\varphi_0 + i\alpha + \frac{(\varphi_0 + k_2\alpha)}{k}\right),$$
$$R_n\sin\left(\varphi_0 + i\alpha + \frac{(\varphi_0 + k_2\alpha)}{k}\right), x, y)$$
$$\omega_n(x,y) = \omega_{n-1}(x,y) \vee_0 \omega_{nx1}(x,y) \vee_0 \omega_{nx2}(x,y) \vee_0 \ldots$$
$$\vee_0 \omega_{nxi}(x,y) \vee_0 \ldots \vee_0 \omega_{nxk^{n-1}}(x,y). \qquad (14)$$

In the previous formulas *k=2, 3, 4, 5, ….*

Pythagorean tree. Pythagoras, proving his theorem, constructed a figure in which squares were placed on the sides of right triangles. If this process is continued, a Pythagorean tree is formed. Using quadratic equations, we construct the equation of the tree, that is

$$\omega_0(a,x,y) = \Big((a^2 - x^2 \ge 0) \wedge_0 \big((b^2 - (y-a)^2 \ge 0) \vee_0 (b^2 - (y+a)^2 \ge 0)\big)\Big) \vee_0$$
$$\vee_0 \Big((a^2 - y^2 \ge 0) \wedge_0 \big((b^2 - (x-a)^2 \ge 0) \vee_0 (b^2 - (x+a)^2 \ge 0)\big)\Big) \ge 0$$

Here $\omega_0(a,x,y)$ - side *2a* and a square whose thickness is 2b.

Using the recursion procedure, we obtain the following [6]: $\omega_n(a,x,y) =$

$$\omega_0(a,x,y) \vee_0 \omega_{n-1}\Big(a\cos(\alpha), \Big(x + a - a\sqrt{2}\cos(\alpha)\sin\Big(\frac{\pi}{4} - \alpha\Big)\Big)\cos(\alpha) +$$
$$+\Big(y - a - a\sqrt{2}\cos(\alpha)\cos\Big(\frac{\pi}{4} - \alpha\Big)\Big)\sin(\alpha), -\Big(x + a - a\sqrt{2}\cos(\alpha)\sin\Big(\frac{\pi}{4} - \alpha\Big)\Big)\sin(\alpha) +$$
$$+\Big(y - a - a\sqrt{2}\cos(\alpha)\cos\Big(\frac{\pi}{4} - \alpha\Big)\Big)\cos(\alpha)) \vee_0 \omega_{n-1}(a\sin(\alpha),$$
$$-\Big(x - a - a\sqrt{2}\sin(\alpha)\sin\Big(\frac{\pi}{4} - \alpha\Big)\Big)\sin(\alpha) + \Big(y - a - a\sqrt{2}\sin(\alpha)\cos\Big(\frac{\pi}{4} - \alpha\Big)\Big)\cos(\alpha),$$
$$\Big(x - a - a\sqrt{2}\sin(\alpha)\sin\Big(\frac{\pi}{4} - \alpha\Big)\Big)\cos(\alpha) + \Big(y - a - a\sqrt{2}\sin(\alpha)\cos\Big(\frac{\pi}{4} - \alpha\Big)\Big)\sin(\alpha)) \qquad (15)$$

here α - The angle of the tree branch when it turns to the left

$0 < \alpha < \frac{\pi}{2}$ takes a value in the interval, the angle of rotation when turning to the right $\frac{\pi}{2} - \alpha$ is equal to

Results and discussion

The Cantor set is a classical fractal, which is generated through an iterative segmentation process. The initial segment [0,1] is taken and the middle third of it is removed at each iteration:

$C_0 = [0,1]$ $C_{n+1} = \frac{1}{3}C_n \cup \frac{2}{3} + \frac{1}{3}C_n$, bunda C_n – n- Cantor set in iteration; Number of segments per iteration 2^n.

Fractal pattern $D = \frac{\log(2)}{\log(3)} \approx 0.6309$. In texture synthesis, the Cantor set is used to: segment a surface; create a pattern structure; and generate texture layers.

A Gilbert curve is a space-filling curve that covers a continuous 2D space. The coordinate recursive representation is: $G(n) = G(n-1) \cup \{R * G(n-1)_{rev} + \vec{v}\}$. This curve has the following properties: it covers all pixels, preserves local proximity, and optimizes the pixel order during texture generation. Therefore, the Gilbert curve is used to determine the sequence of texture pixel generation.

The Gosper curve is generated by a fractal L-system. Initial rules: A, B. Production rules: A→A−B−−B+A++AA+B−

B→+A−BB−−B−A++A+B,

"+" — 60° turn left;

"-" — 60° turn right.

As the iteration increases, the curve becomes more complex. In texture synthesis, the Gosper curve is used to create: a complex fractal pattern; a natural look; and multi-layered textures.

The extension of fractal lines to a 2D texture is done as follows.If the coordinates of the curve are: $P = (x_i, y_i)$, If so, the texture value is: $T(x,y) = f(x,y) + \alpha F(x,y)$, in this $f(x,y)$ − basic texture, $F(x,y)$ − fractal pattern; α − intensity parameter. In 3D space: $T(x,y,z) = f(x,y,z) + \beta F(x,y,z)$, This allows you to generate material textures such as stone, wood, and marble.

A fractal texture consists of several layers.: $T = T_1 + \omega_2 T_2 + \omega_3 T_3$, in this T_1- basic texture, T_2- fractal pattern, T_3- noise.

Texture generation works based on the following algorithm: determining the texture size; generating a fractal curve; calculating the curve coordinates; calculating the texture value for each coordinate; generating pixel color; layering the texture; creating the final texture image, Figure 1.

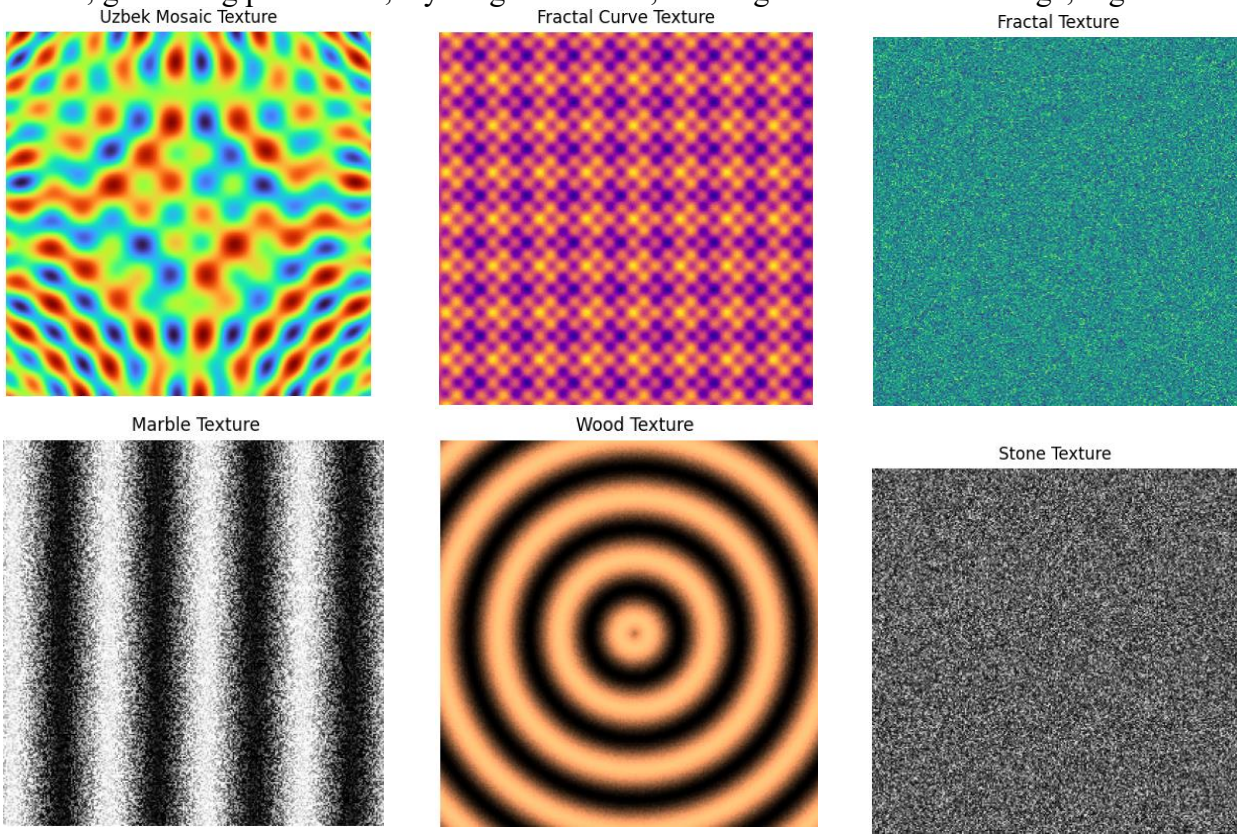

Figure 1. Texture generation results

The results of the study showed that fractal-based textures provide a high level of complexity

and naturalness. The Cantor set was effective in segmenting surfaces and creating the structural basis of patterns. Complex patterns were generated on surfaces using Gilbert and Gosper curves. Since these curves have a space-filling property, the texture elements are arranged smoothly and continuously. Procedural textures were found to reduce memory consumption and provide flexible control through parameters.

Conclusion

According to the results of the study, procedural texture synthesis based on the Cantor set and Gilbert and Gosper curves is an effective tool for 3D reconstruction of historical objects. The possibilities of using procedural texture synthesis methods based on the Cantor set and Gilbert and Gosper curves in 3D reconstruction of historical objects were studied. The results of the study showed that it is possible to create complex, non-repeating and realistic-looking textures on surfaces using fractal geometry elements. The Cantor set played an important role in forming the fractal segmentation of the texture structure, and the Gilbert and Gosper curves allowed generating patterns that cover surfaces continuously. The methodology developed during the study demonstrated the effectiveness of procedural texture synthesis. It was found that textures created based on fractal algorithms are much more economical in terms of memory consumption than traditional image textures. In addition, the possibility of parametric control made it easier to model the surfaces of various historical materials. This allows for a more accurate and aesthetically perfect depiction of historical objects in a virtual environment.

The results of the study showed that procedural texture synthesis works effectively even in real-time graphics systems. It was found that it is possible to dynamically generate textures using shader technologies. This expands the prospects for using this method in virtual reality, augmented reality, and interactive visualization systems. This approach allows for the automatic restoration of damaged or lost texture parts of historical objects. At the same time, it is also possible to significantly optimize the process of processing large volumes of 3D cultural heritage data. In addition, the integration of textures created based on fractal algorithms with various graphics engines (Unity, Unreal Engine) is also a promising direction. This will expand the possibilities of creating high-quality visual models in virtual museums, educational platforms, and digital cultural heritage projects. As a result, procedural texture synthesis can become one of the important technological tools for the digital preservation of historical objects and their presentation to the general public.

References

1. Ebert, D. S., Musgrave, F., Peachey, D., Perlin, K., & Worley, S. (2003). Texturing and modeling: A procedural approach. Morgan Kaufmann. https://doi.org/10.1016/B978-155860848-1/50002-3
2. Lagae, A., Lefebvre, S., Drettakis, G., & Dutré, P. (2010). Procedural noise using sparse Gabor convolution. ACM Transactions on Graphics, 29(4). https://doi.org/10.1145/1778765.1778803
3. Akenine-Möller, T., Haines, E., & Hoffman, N. (2019). Real-Time Rendering (4th ed.). CRC Press. https://doi.org/10.1201/9781315368934
4. Shirley, P., & Marschner, S. (2015). Fundamentals of Computer Graphics. CRC Press.
5. Musgrave, F. K. (2012). Fractal terrains and procedural textures. Computer Graphics and Applications.
6. Cook, R. L. (2005). Stochastic sampling in computer graphics. ACM Transactions on Graphics.
7. Mandelbrot, B. (2004). Fractal geometry of nature. W. H. Freeman.
8. Peitgen, H. O., Jürgens, H., & Saupe, D. (2006). Chaos and fractals. Springer.
9. Lefebvre, S., & Hoppe, H. (2006). Appearance-space texture synthesis. ACM Transactions on Graphics.
10. Smelik, R., Tutenel, T., Bidarra, R., & Benes, B. (2014). Procedural generation of terrains for games. ACM Transactions on Multimedia Computing.

11. Beknazarova, S., Abdullayev, S., Abdullayeva, O., & Abdullayev, Z. (2024). Machine learning method for predicting human movements. AIP Conference Proceedings, 3244, 030036. https://doi.org/10.1063/5.0242100
12. Beknazarova, S., Abdullayeva, O., Abdullayev, S., & Abdullayev, Z. (2024). Image processing: Face recognition by neural networks. E3S Web of Conferences, 587, 01010. https://doi.org/10.1051/e3sconf/202458701010

С. С. Бекназарова*, К. К. Бекназарова
*т.ғ.д., профессор, saida.beknazarova@gmail.com, Мұхаммед әл-Хорезми атындағы Ташкент Ақпараттық технологиялар университеті, Ташкент, Өзбекстан
зерттеуші, k.beknazarova@gmail.com, Мұхаммед әл-Хорезми атындағы Ташкент Ақпараттық технологиялар университеті, Ташкент, Өзбекстан

3D ҚАЙТА ҚҰРУДАҒЫ ТЕКСТУРАЛАРДЫҢ ПРОЦЕДУРАЛЫҚ СИНТЕЗІ

Түйін

Мақалада тарихи нысандарды 3D қайта құруда және виртуалды ортада жоғары сапалы текстураларды жасауда процедуралық текстураны синтездеудің артықшылықтары қарастырылады. Кантор жиынтығы, Гилберт және Госпер қисықтары сияқты фракталдық және кеңістіктік қисықтарды қолдана отырып, бірегей және шынайы текстураларды құру әдістеріне назар аударылады. Бұл жұмыстың негізгі үлесі текстураларды жасау үшін фракталдық геометрияны және кеңістікті толтыратын қисықтарды жүйелі түрде қолдану болып табылады. Әдістеме үш негізгі математикалық құрылымды мұқият зерттейді: үзік-үзік және шаң тәрізді үлгілерді жасауға арналған Кантор жинағы, локализацияны сақтай отырып, екі өлшемді кеңістікте қозғалуға арналған Гильберт қисығы және алтыбұрышты құрылымдар мен қар ұшқынына ұқсас құрылымдарды жасауға арналған Госпер қисығы. Бұл қисықтарды параметрлеу және оларды фракталдық Шу функцияларымен біріктіру арқылы авторлар жұмыс уақытында текстураларды синтездейтін гибридті алгоритмдік құбырды ұсынады, бұл әртүрлі масштабтау деңгейлерінде шексіз әртүрлілік пен мінсіз бөлшектерді қамтамасыз етеді (бөлшектерге тәуелсіз синтез).

Кілттік сөздер: Текстураның процедуралық синтезі, кантор жиынтығы, Гильберт қисығы, Госпер қисығы, фракталдық технология, 3D қайта құру, виртуалды орта, компьютерлік графика

С.С. Бекназарова*, К.К. Бекназарова
*д.т.н., профессор, saida.beknazarova@gmail.com, Ташкентский университет информационных технологий имени Мухаммада Аль-Хорезми, Ташкент, Узбекистан
исследователь, k.beknazarova@gmail.com, Ташкентский университет информационных технологий имени Мухаммада Аль-Хорезми, Ташкент, Узбекистан

ПРОЦЕДУРНЫЙ СИНТЕЗ ТЕКСТУР В 3D-РЕКОНСТРУКЦИИ

Аннотация

В статье рассматриваются преимущества процедурного синтеза текстур при 3D-реконструкции исторических объектов и создании высококачественных текстур в виртуальных средах. Основное внимание уделяется методам создания уникальных и реалистичных текстур с использованием фрактальных и пространственных кривых, таких как набор Кантора, кривые Гилберта и Госпера. Основной вклад этой работы заключается в систематическом применении фрактальной геометрии и кривых, заполняющих пространство, для создания текстур. В методологии тщательно изучаются три фундаментальные математические конструкции: набор Кантора для создания прерывистых и пылевидных узоров, кривая Гильберта для перемещения по двумерному пространству с сохранением локальности и кривая Госпера для создания шестиугольных структур и структур, похожих на снежинки. Параметризуя эти кривые и интегрируя их с функциями фрактального шума,

авторы предлагают гибридный алгоритмический конвейер, который синтезирует текстуры во время выполнения, обеспечивая бесконечное разнообразие и безупречную детализацию на разных уровнях масштабирования (синтез, не зависящий от деталей).

Ключевые слова: процедурный синтез текстур, множество Кантора, кривая Гильберта, кривая Госпера, фрактальные технологии, 3D-реконструкция, виртуальная среда, компьютерная графика