УДК 004

O.V. Burdeinyi
O.I. Chernyak
S.O. Kot

Visualization of complex three-dimensional environments using Raymarching and procedural modelling

Vinnytsia National Technical University

Анотація

Розроблено програмне забезпечення в середовищі Unity з використанням HLSL-шейдерів, яке спрямоване на використання продуктивних методів візуалізації тривимірних складних динамічних сцен методом Raymarching. Запропонований підхід забезпечує багатопотокове високопродуктивне виконання обчислень, що дозволяє покращити графічне відтворення зображень у реальному часі.

Ключові слова: полігон, піксель, модель, динамічність, шейдер, SDF, HLSL, Raymarching, Unity.

Abstract

The sofrware was developed in the Unity environment using HLSL shaders aimed at designed to utilize efficient methods for rendering complex three-dimensional dynamic scenes using the Raymarching method. The proposed approach provides multi-threaded high-performance computing, which improves real-time image rendering.

Keywords: polygon, pixel, model, dynamism, shader, SDF, HLSL, Raymarching, Unity.

Introduction

The current trend in microprocessor development reveals a significant decrease in the rapid growth of computing power [1]. At the same time, the expansion of the usage scope of graphics software requires more and more resources, initiating a necessity of alternative solution search. The most common shared method of model visualization implements rendering based on wireframe models, in which an object is represented by a mesh of vertices and edges [2]. This method is used in combination with surface rendering [3], which fills in the empty gaps between the edges by forming and coloring conditional polygons to simulate three-dimensionality. In fact, all such models use the rasterization method [4], in which each polygon is projected onto the screen plane. After that, the pixels that the polygon covers are determined. As a result, this gives the output color of the pixel. The research presented by the authors offers an alternative method for calculating and visualizing complex three-dimensional models in dynamic real-time scenes, which makes it much easier to find the solution of the problem of processing highly detailed objects in real time, since existing methods cannot perform it due to the large amounts of vertex and edge data.

Main section

Within the scope of this research, software and an HLSL shader [5] were developed for real-time three-dimensional visualization of complex dynamic models. Since in the proposed approach most of the calculations are performed by the graphics shader, the main load falls on the graphics processor rather than the central processor, which provides better performance through the use of multithreaded task execution.

The idea behind the proposed method is to replace traditional polygons with vertices and edges with a combination of symbolic distance functions of simple shapes [6] with additional noise to create composite functions of complex models and their subsequent visualization using a method similar to rasterization, called Raymarching. Unlike rasterization, Raymarching represents the passage of imaginary rays with a certain movement step, which will pass through coordinates converted from the screen to three-dimensional space. Geometric figures will then be placed in this space. At the same time, each pixel will also be coloured depending on the result of the passage, intersection with the surface of the figure or lack of intersection.

Since calculations are performed for each pixel on the screen, to ensure smooth calculations and reduce visualization delays, it is recommended to use a shader written in HLSL. This language makes maximum use of the multi-threading advantage of the computer's graphics processor compared to the central processor, which is used only to form and transfer data to the shader for further calculations.

The design of composite and modified distance functions for future models is based on the use of Raymarching calculation properties and additional techniques to ensure detail. Each distance function has a input parameters points in coordinate space with certain dependent parameters of the figure, and returns numerical value that indicate the distance to the surface of a specific figure relative to a specified point in space. This makes it possible to perform operations such as inion shapes, calculating their difference and intersection, as shown in Figure 1.

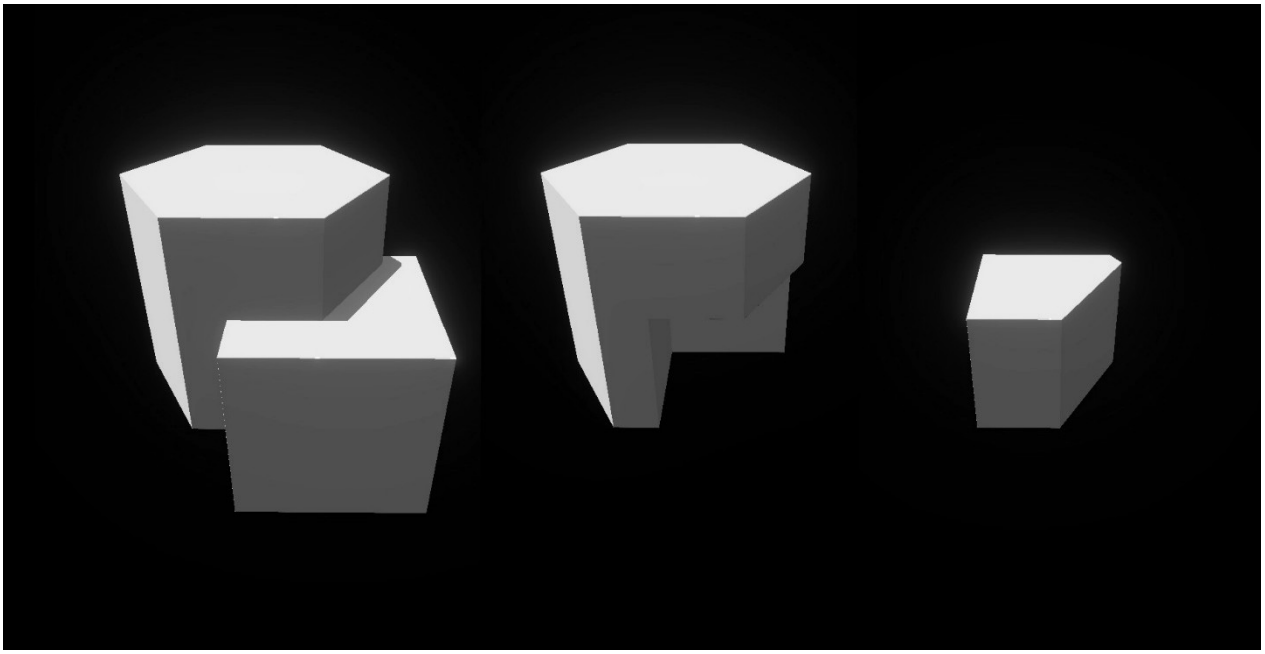

Figure 1 – The result of performing operations on cube and hexagonal prism figures

Here are the results of the union, difference, and intersection operations are presented. Each of these operations is performed by executing mathematical formulas such as min, max, and max with a negative parameter, respectively (formulas 1.1-1.3, respectively).

$$\text{Union}(x)=\min(d1(x), d2(x)), \quad (1.1)$$

$$\text{Inter}(x)=\max(d1(x), d2(x)), \quad (1.2)$$

$$\text{Diff}(x)=\max(d1(x), -d2(x)), \quad (1.3)$$

Furthermore, since three-dimensional space is represented by numerical values relative to the screen space, it becomes possible to apply scaling, rotation, duplication, and shift (formulas 1.4-1.8, respectively).

$$d(x)\rightarrow s\cdot d(x/s), \quad (1.4)$$

$$d(x)\rightarrow d(R^{-1}x), \quad (1.5)$$

where R — is the rotation matrix;

$$x\rightarrow \text{mod}(x,p)-p/2, \quad (1.6)$$

where p — repetition period;

$$d(x)\rightarrow d(x-t), \quad (1.7)$$

where t— displacement vector.

Extra methods for adding three-dimensional relief and giving graphic objects a natural look include applying fractal and pearl noise. In addition, for complex geometry combinations, smoothed Boolean operations are used with a numerical smoothing parameter, which provides more flexible control between sharp and smooth geometry transitions.

Despite writing the shader in HLSL, the C# programming language was chosen for the main program. Its advantages are OOP orientation and ease of use. Additioally, this language is convenient for creating an interactive user interface.

The development perspective of the proposed approach include the implementation of this software as a graphical basis for applications with a high demand for highly detailed dynamic and static scenes..

Conclusions

The work provides an alternative method for visualizing complex detailed models of objects based on Raymarching using basic shapes and numerical operations on their geometry. This method will reduce the load on application system components with high-quality high-speed graphics.

REFERENCES

1. The End of Moore's Law and Why Game Developers Care / Anastasiia Matrokhina. – URL: https://www.cn.ua/news/apple/7269-zakinchennia-zakony-myra-i-chomy-rozrobnikam-igor-ne-vse-odno.html (accessed: 22.10.2025).
2. Romanyuk V. A. Computer Graphics: Textbook. – Vinnytsia: VNTU, 2010. – URL: https://web.posibnyky.vntu.edu.ua/fitki/8romanyuk_komp_grafika/zmg1/zmg/31.htm, sec. 3.1 "Representation of Surface by Polygonal Mesh" (accessed: 22.10.2025).
3. Azarov O. D., Dudnyk O. V., Shvets S. I. Fundamental Principles of Physically Correct Rendering. Proceedings of the Scientific-Practical Conference. – Vinnytsia: VNTU, 2023. URL: https://ir.lib.vntu.edu.ua/bitstream/handle/123456789/41917/20288.pdf (accessed: 22.10.2025).
4. Rasterization: a Practical Implementation – Scratchapixel. – URL: https://www.scratchapixel.com/lessons/3d-basic-rendering/rasterization-practical-implementation/overview-rasterization-algorithm.html (accessed: 22.10.2025).
5. Raymarching Distance Fields / Inigo Quilez. – URL: https://iquilezles.org/articles/raymarchingdf/ (accessed: 22.10.2025).
6. Writing HLSL shader programs – Unity Manual. – URL: https://docs.unity3d.com/6000.2/Documentation/Manual/writing-shader-writing-shader-programs-hlsl.html (accessed: 22.10.2025).
7. Unity Documentation. – Веб-сайт компанії Unity Technologies. – URL: https://docs.unity.com/en-us (accessed: 22.10.2025).

Бурдейний Олег Володимирович ― студент групи 1КІ-25м, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Вінниця, e-mail: burdeyniy.007.com@gmail.com

Черняк Олександр Іванович – кандидат технічних наук, доцент кафедри обчислювальної техніки Вінницького національного технічного університету, Вінниця, e-mail: chernyak@vntu.edu.ua

Кот Сергій Олександрович – кандидат технічних наук, доцент кафедри іноземних мов Вінницького національного технічного університету, Вінниця, e-mail: kot.sergii@vntu.edu.ua

Burdeinyi Oleh Volodymyrovych. ― student of group 1CI-25m, Faculty of Information Technology and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: burdeyniy.007.com@gmail.com

Chernyak Oleksandr Ivanovych — PhD, associate professor of the Department of Computer Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: chernyak@vntu.edu.ua

Kot Serhiy Oleksandrovych – PhD, associate professor of the Department of Foreign Languages, Vinnytsia National Technical University, Vinnytsia, e-mail: kot.sergii@vntu.edu.ua