Eng Int Syst (2024) 5: 411–419

Engineering
Intelligent Systems

Animation Design System Based on 3D Image Technology

Fan Jiang*

Major in Cartoon and Animation, Department of Film Press and Contents, Cheongju University, Chungcheongbuk-do, (28503), Republic of Korea

With the rapid development of computer technology, imaging technology, in particular the 3D (three-dimensional) modeling and simulation techniques are being increasingly used. However, due to the complexity of the 3D mapping technique itself, in the development of new 3D mapping technology as well as in the implementation in the engine, how to produce a 3D drawing has become difficult in today's design and drawing domains. Therefore, based on the existing 3D mapping techniques, an in-depth analysis of the characteristics of the existing 3D drawing techniques is carried out, and a drawing system based on 3D technology is proposed. Also, in this study, the engine structure is divided, and the architecture and functional design of the 3D engine animation system is completed. Regarding the 3D image engine, the analysis of the animation design system indicated that when the HDR (High-Dynamic Range) effect is on and off, the rendered frame rates averaged 133.76FPS and 233.44FPS respectively. Therefore, a study of the animation design system is necessary.

Keywords: Animation design system, three-dimensional image, rendering effect, high dynamic range

1. INTRODUCTION

Computer-aided drawing (CAD) makes use of mathematical methods to convert 2D (two-dimensional) or 3D (three-dimensional) images into a raster format, and the result can be displayed on the computer screen so that the image is visually realistic. Among these mathematical methods of computer-aided drawing, 3D mapping technology integrates a large amount of computer technology knowledge, and a solid discipline foundation is the prerequisite for the rapid development of 3D application software. Currently, 3D mapping can be easily developed with OpenGL and DirectX technologies with satisfactory results but, in practice, it is difficult for developers to master these techniques. Both OpenGL and DirectX are non-object-oriented, offering no dedicated model tools and only basic graphics elements. For complex images and scenes, drawing with basic elements is very troublesome, and programming is very difficult. In addition, complex operations such as effect rendering, dynamic tailoring, and custom coloring all need to be written from scratch. Therefore, in order to reduce the work of 3D graphics application, it is necessary to develop a 3D graphics engine with efficient functions, isolating the underlying hardware operation, and simple graphics processing [1]. The 3D engine is the key component of 3D image processing software, which separates developers from the underlying hardware, graphics algorithms, and facilitates the development. This paper focuses on the development and implementation of a 3D graphics engine, focusing on the development of 3D animation, and provides the corresponding interface for the production of 3D animation.

At present, the design of animation systems is one of the popular research topics in the animation field. Among them, Tiantada proposed Gemini, which is a declarative syntax and recommendation system for the animation conversion between single-view statistical graphics [2]. Zhang, based on his research of the UE 4 engine, proposed a 3D animation automatic generation technology based on the image engine [3]. Fabrizio provides an add-on to the Blender animation suite that enables users to switch between a native (Virtual

*Address for correspondence: Fan Jiang, Major in Cartoon and Animation, Department of film press and contents, Cheongju University, CheongJu-Si, Chungcheongbuk-do, (28503), Republic of Korea, Email: hf470330825@163.com

Reality, VR) based and immersive interface, and use the latter to perform a representative set of task [4]. However, due to the lack of data sources, these researches address only the theoretical aspects of the technology, and not the practical applications.

It is very worthwhile to study animation design systems. Wolfgang proposed a new hybrid animation framework, which uses the latest advances of deep learning to provide an interactive animation engine, which can be used for facial expression editing [5] through simple and intuitive visualization methods. Ngiik analyzed the application of 3D hologram technology in junior school students to find the learning effect of 3D hologram technology in the classroom [6]. According to Kumar, the purpose is to determine the conditions under which animation-based teaching methods will lead to better learning results [7]. However, due to the traditional thinking about teaching methods, the two cannot be integrated to reap the advantages of the technology.

In this paper, we design and implement the 3D graphics engine animation system, and discuss the key technologies in the engine system. The functions of the 3D graphics engine animation system are re-divided and the architecture is designed accordingly. Finally, the functional components and overall architecture are redesigned. Based on programmable graphics hardware, high dynamic range images (High-Dynamic Range, HDR) effects are realized. This function is integrated into [8–9] in the 3D graphics engine animation system as a lighting effects module. We show the soft shadow algorithm of surface light source with quadrilateral approximation. The idea of the algorithm is to map light models of other shapes into surface sources of quadrangle approximation. The experimental verification and analysis indicate that the performance of the algorithm has been greatly improved and can meet the requirements of real-time performance of the system.

2. RESEARCH ON THE 3D-BASED ANIMATION DESIGN SYSTEM

2.1 3D Graphics Engine Analysis

The development of graphic application software together with 3D drawing technology requires numerous technologies and algorithms, and there are some difficulties in terms of practical application. Many open-source architectures, such as OpenGL and Direct3D, provide an application programming interface (Application Programming Interface, API) for graphics development. However, there is an urgent need for software developers to package their underlying graphics development software to provide a software architecture that can facilitate software development. This paper explains in detail the overall design of the 3D graphics engine and its key modules [10].

(1) Performance analysis of the engine

Because the computer 3D graphics engine is based on computer software, its design process is more complicated and involves many aspects. Therefore, when building the 3D engine, we need to conduct a detailed analysis of various aspects, including the implementation of functions, the provision of services, and the relationship of various modules. The 3D engine is the core of 3D graphics, and its design concept is critical to its success. The ductility of the interface must be fully considered for future modification and upgrade. Of course, when designing the engine, attention must be paid to the independence and relevance of the system. The optimal architecture of the system does not depend on the operating system, can be transplanted on any platform, and has good compatibility, not constrained by compiler, hardware facility version, parameters, etc.

'Up' requires a strong API interface to support higher-level 3D graphics applications, and 'down' is the operating system API and the graphics API. The underlying package is critical, and the goal is to completely separate the engine system from the operating system and hardware. In the analysis of project requirements, the functional design of the engine should focus on the design and analysis, especially according to the software engineering method: the modular method of design and construction. In this process, the division of each module should be undertaken carefully, and the connection between each module should be close. As shown in Figure 1.

Based on the object-oriented design idea, 3D graphics engine system has better readability and clearer semantics, so that it has better expansion ability. In terms of the development cycle, it is easy to manage, low development cost, low maintenance cost, and more importantly, the engine is very portable and easy to use. The object-oriented concept is based on the needs of users and the development environment. Although the system cannot fully meet all the requirements of users, the three-dimensional engine animation system proposed in this paper covers the basic functions of the whole engine. Among them, scene management, rendering management, resource management and other functional modules are the key components of the proposed system.

The 3D graphics engine animation system developed in this paper was summarized and abstracted the classes, reducing the complexity of the relationship between the components. Multiple classes are used to create multiple interfaces, thus reducing the complexity of the program, hiding the basic realization, the design of the interaction interface between the modules reduces the complexity of the relationship between the modules. The interface provides a direct, intuitive and single means of interaction between the modules, facilitating their development and management.

(2) The 3D graphics engine system design

Graph-rendering pipeline technology is the basis of real-time graph drawing. The technology is used to transform three-dimensional objects into two-dimensional objects through special processing, so as to adapt to the output of display hardware, and make a series of transitions from 3D to 2D. In practice, a fixed functional pipeline and a programmable pipeline are usually used to draw the 3D images.

A fixed assembly line consists of certain steps. Each step has a clear definition, playing a different role in the assembly line. Each stage has an input and output, and

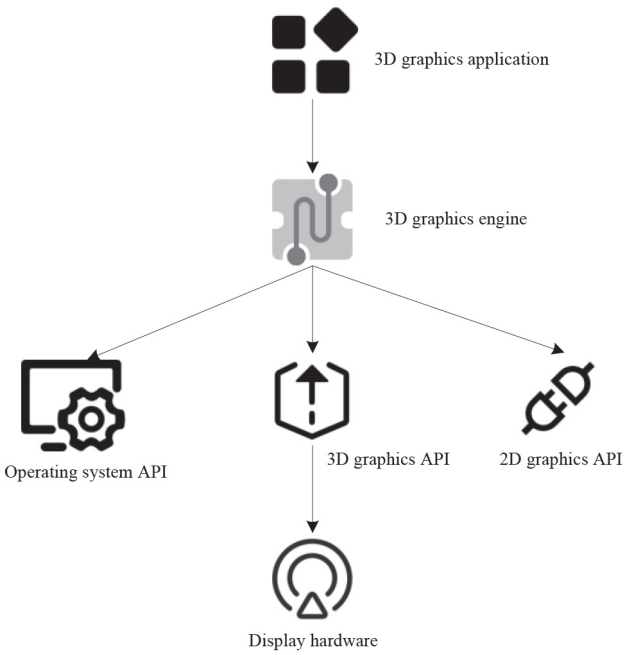


Figure 1 3D graphics, engine system position.

the output of the previous step is the output of the next step. Its contents include: vertex conversion and illumination, combination of elements and rasterization, interpolation, mapping and coloring, raster manipulation of four steps, these steps complement each other, one is indispensable. Of these, vertex transformation and illumination are the main processing steps. When the 3D vertex is sent to the graphics card, the coordinate system of the vertex is converted into the frame system of the picture by digital operation, so that the texture frame is generated on the graphics processor (Graphics Processing Unit, GPU); in the second stage, the elements and sterilization process are executed. The raw data sequence is first taken as input and combined with the corresponding information to transform it into a series of basic geometries of points, segments, triangles, etc. as shown in Figure 2.

Each element has its own characteristics. After calculation and cutting, some pixels are removed to become a visible flat cut or section. After rasterization processing, the element information becomes a unit and a collection of pixels; finally, the resulting data is input into the third level of the image for interpolation, mapping and coloring. This stage is an important link in the pipeline processing of image fixed function. When a specific element attribute is required, interpolation will appear, and the color of each element is determined by mapping and color processing. Of course, in practice, the correctness of the element must be determined to avoid additional frame cache updates. A test should be conducted to determine whether the unit is valid; any unqualified unit is discarded. The cells that pass the test are retained, and then the cells and the corresponding pixel information are merged. Lastly, the frame cache write operation is used to complete the fixed pipeline drawing of the graph.

The structure of the 3D graphics engine is the core of the whole 3D graphics engine, so it should be considered comprehensively in the architecture design of the whole system. Its specific functions, relationships between modules, portability, scalability and so on must be considered. Based on this, the development scheme [11] based on 3D graphics engine is proposed. The animation system of the engine consists of two core modules: the engine kernel and the application expansion, as shown in Figure 3.

The system is located in the middle of the application and graphics API, which is compatible with the OpenGL library and DirectX3D interface. The extendable application programming interface is abstract and is used for user invocation. The core functions of the system include: scene management, rendering management, resource management, among others.

The 3D graphics engine animation system designed in this paper is intended for the packaging of a basic graphics operation, and offers a set of standardized functional interfaces to provide a solid structured support platform for high-level applications and facilitate the development of upper application software, Through the shading of underlying graphics drawing, direct interaction with hardware can be avoided, and a series of graphics processing and rendering techniques is extracted based on OpenGL and DirectX3D. In order to achieve the underlying basic operation, several aspects must be taken into account: the network communication, data drive, database technology, data synchronization and other technologies.

Database operation technology is an essential link in the software development process and a problem that calls for an urgent solution. In order to achieve better access to the database, an object-oriented 3D graphics engine animation system is proposed and encapsulated to operate [12–13]. Also, the network communication is a practical problem that each software system should consider. Whether data-driven or data-synchronized, interaction with the hardware is required, and this can be done by technology that provides ordinary drawing tools.

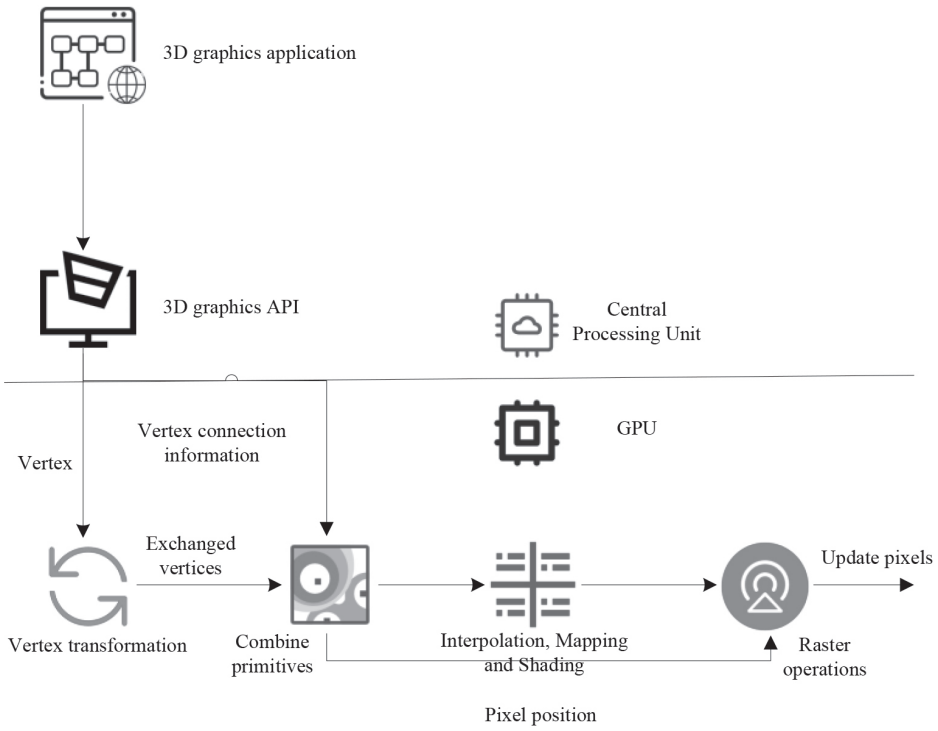


Figure 2 3D graphics engine rendering fixed pipeline.

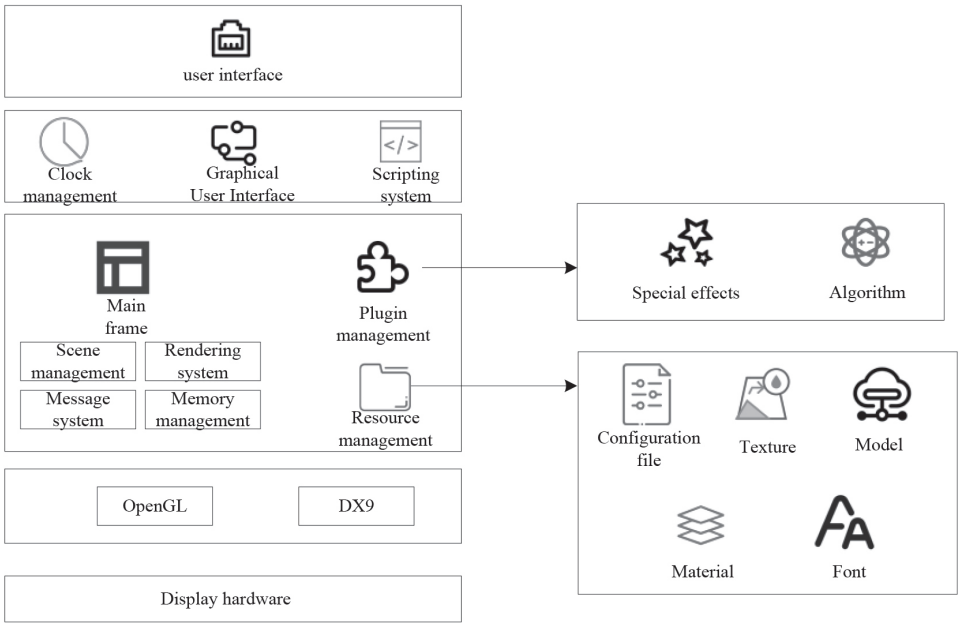


Figure 3 3D graphics engine animation system architecture.

The core modules of the system include: main framework, scene management, rendering management, resource management, information processing; the core modules of the extended application layer are: clock management and basic graphics unit management.

2.2 HDR Implement the Framework

(1) Tone-color mapping

To address the disadvantage of the too narrow range of brightness dynamic of conventional devices, reconstruction of images with limited dynamic range is required. In this study, the dynamic range of HDR images is narrowed down in several ways.

Tonal mapping (or color scale reconstruction) can compress or map the dynamic range, so that a high dynamic range image can be displayed on a conventional display device. The early color mixing algorithm is relatively simple, generally used only to compress the brightness, set the largest and minimum dynamic range window, in order to achieve the purpose of HDR image compression. Recently, research on color mapping has focused not only on the largest dynamic range, but also on the visual quality of HDR images so as to

Table 1 Reality enhancement function and rendering performance comparison.

Types	Engine of this Article	Threejs	Babylonjs
Dynamic Shadows	Have	Have	None
Ray Tracing	Have	Have	Have
Shadow Mapping	Have	Have	None
Light Source Rendering	Have	None	Have
Texture Map	None	Have	None
Single Object Rendering Frame Rate	46FPS	46FPS	44FPS
Multiple Object Rendering Frame Rate	28FPS	37FPS	42FPS
Average Model Loading Speed	764ms	562ms	433ms

improve the display effect of HDR images and maximize the dynamic brightness.

In regard to graphics hardware, it is necessary to achieve both programmable and high-quality drawing. The current mainstream Shader graphics accelerator has a high level of hardware, and is greatly improved in terms of the number of instructions, memory, material, and two-dimensional texture clarity. The HDR effects produced in this study are based on Shader3.0 graphics acceleration card, so Shader3.0 graphics acceleration card is used on the graphics hardware.

Two factors can determine the effect of a graphics presentation: 1) the clock of the graphics processor, and as the clock frequency increases, the graphics card processing works faster; 2) the storage space of the display card and the clock frequency of the bus. The focus of this paper is on the middle and high-end imaging accelerator, using NVIDIAGeforeGTX780 to complete the process.

HDR special effects are a set of 3D graphics engine animation [14] with special effects as the core. It offers developers an advanced special effect that can be called upon when required.

(2) Engine comparison

At this point, the engine has completed its basic functions, including core rendering, scenes, cameras, material textures, basic geometry, math, tools, and more. Its performance is very similar to the now-popular 3D engines, Threejs and Babylonjs, albeit with greater realism.

Before the release of the WebGL standard, all engines were based on DirectX and OpenGL, which limited the diversity of platforms. The 3D engine, developed by the Key Laboratory of Information Technology of the Ministry of Education, is still the basic rendering basis of Silverlight, and Microsoft has abandoned the technology. The base part of WebGL is still OpenGL, which can run in WebGL without an additional browser plug-in. In addition, you can create the 3D scenes just by writing the code. It is clear that WebGL works well for the development of 3D graphics software, such as 3D roaming and online games, as well as multi-platform extensions.

There are already numerous open-source engines based on WebGL development, such as Threejs and Babylonjs, which have their own features. Below, Threejs and Babylonjs are used as references to compare the engines described in this study. Table 1 shows the comparison results.

3. PERFORMANCE TEST AND ANALYSIS OF 3D GRAPHICS ENGINE ANIMATION DESIGN SYSTEMS

3.1 Comparison Between 3D Graphics Engine Animation System and Other Mainstream Engine Systems

Object-oriented graphics rendering engine (Object-oriented Graphics Rendering Engine, OGRE) and open Scene diagram (Open Scene Graph, OSG) are the current mainstream graphics engines. When designing and implementing the animation of the 3D graphics engine, this study also integrates some of their experiences and design results. This section compares the dynamics of the current OGRE and the OSG 3D graphics engines.

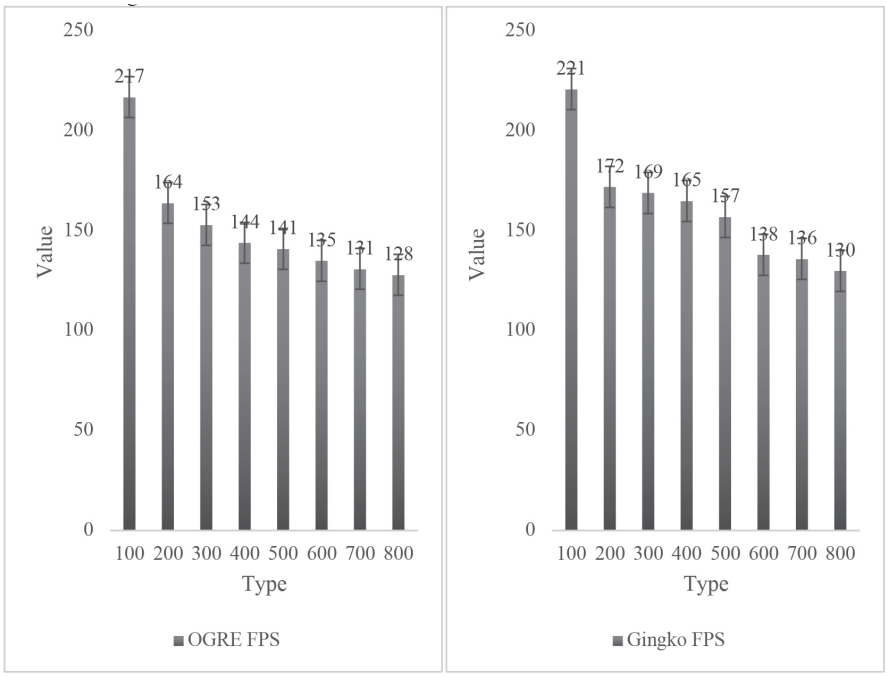
The OGRE is a scenario-oriented, very flexible 3D engine, developed in the C++ language. It makes it easier and more direct for developers to use the development of the 3D drawing system, which hides the specific implementation process of the underlying system library and provides a set of interface [15–16] for the application layer.

The OSG is an open-source cross-platform 3D drawing engine that allows programmers to easily create high-performance cross-platform interactive drawing software. The software uses OpenGL technology to provide a series of application interfaces on the C++ platform. It is a middleware that provides advanced drawing capabilities and various organizational functions, so that the development team does not need to implement and optimize the underlying graphics.

In this paper, we propose a 3D drawing engine based on C++ technology (which refers to the Gingko system). The system provides the main functions of the main graphics engine as well as an underlying graphics development API. It allows developers to use the OpenGL and the Direct3D API to draw the images.

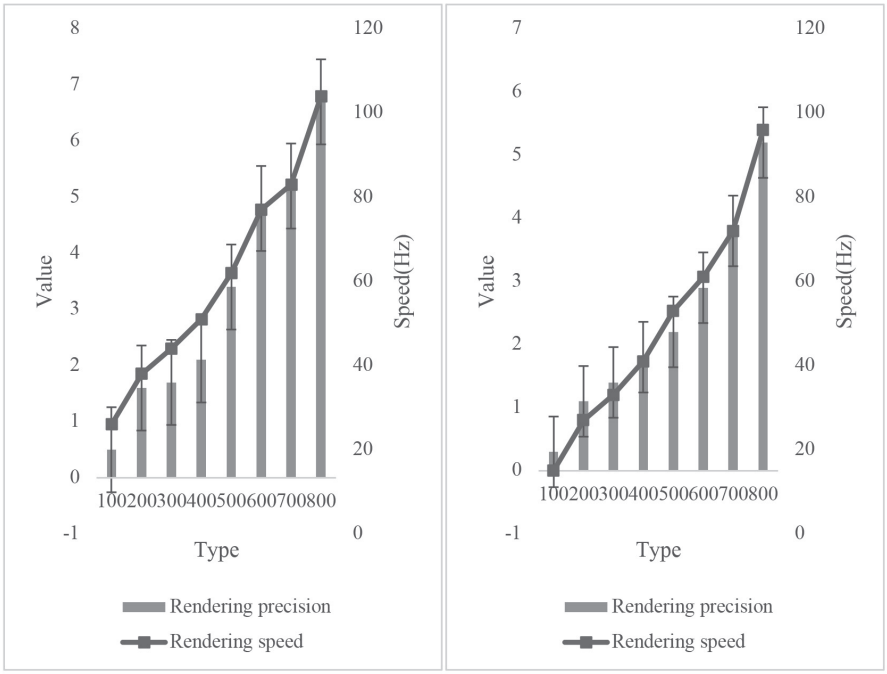
The comparison of the frame rate of the 3D graphics engine animation system and the frame rate of the OGRE engine is shown in Figure 4.

As shown in Figure 4 (a), when the number of triangular sheets is 100, the frame rate is the highest and the highest is 217 FPS, and when the number of triangular faces is 800, the frame rate is the lowest and the lowest is 128 FPS. In Figure 4(b), when the number of triangular faces is 100, the highest frame rate is 221 FPS, and when the number of triangular faces is 800, the frame rate is lowest and the lowest is 130 FPS.



(a). Frame rate of the OGRE (b). The frame rate of the Gingko

Figure 4 OGRE and Ginko rendering frame rate.



A.OGRE B.Gingko

Figure 5 OGRE and Gingko rendering accuracy and rendering speed statistics diagram.

Hence, the effect of both methods is equal when the number of triangular sheets is small. However, when the number of triangles increases, the effect of these two methods decreases. When the number of triangular images to draw exceeds 600, the drawing efficiency of both methods tends to stabilize, while the 3D image engine is much faster than the OGRE algorithm. This was due to the initial conceptual design of the system, the goal of which is to develop a 3D image processing software with high real-time performance.

This paper is tested on a personal computer (Personal Computer, PC). Figure 5 allows a comparison of the rendering accuracy and rendering speed of the 3D graphics engine animation system and those of the OGRE engine.

As Figure 5 shows, in A, the rendering accuracy of OGRE reaches a maximum at the number of factangles of 800, with a maximum of 6.7. When the number of triangular faces is 100, the rendering accuracy is the minimum, and the minimum value is 0.5. The rendering speed of OGRE

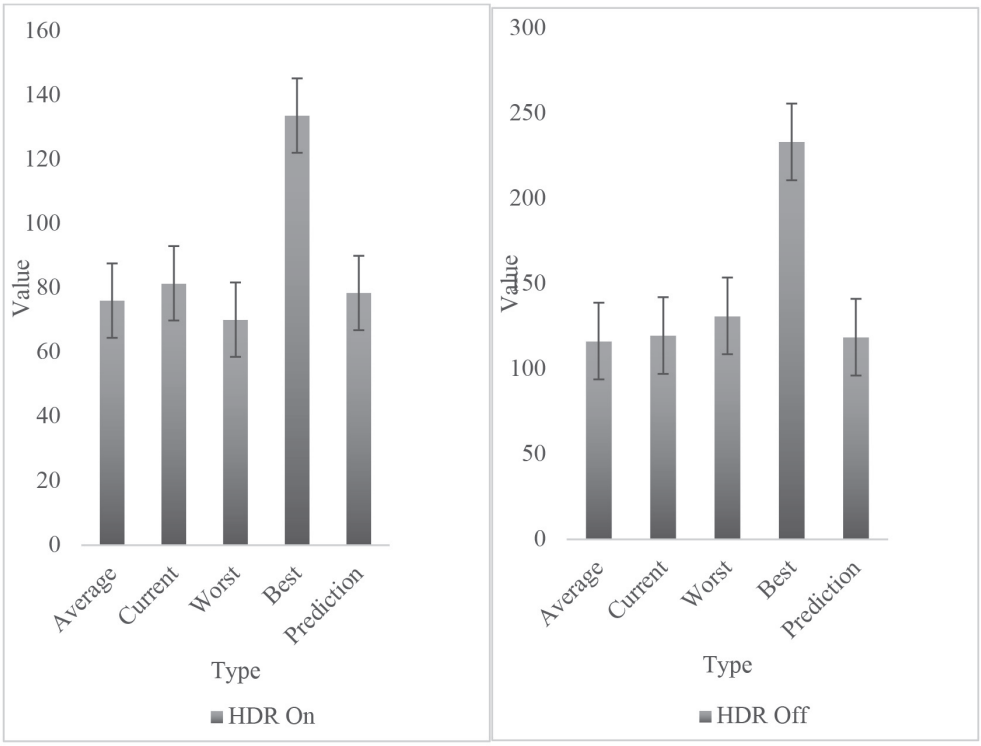


Figure 6(a). HDR open

Figure 6(b). HDR close

Figure 6 Comparison of frame rates with HDR in 'on' and 'off' mode.

reaches the maximum at 800, the maximum value is 104Hz, the speed is the minimum at the number of triangular faces is 100, and the minimum value is 26Hz. In B, the rendering accuracy of Gingko reaches a maximum number of 800 with a maximum of 5.2, and a minimum number of 100 with a minimum of 0.3. The rendering speed reaches the maximum when the number of triangular faces is 800, the maximum value is 96Hz, the number of triangular faces is 100, and the minimum value is 15Hz.

3.2 HDR Module Special Effects and Soft and Hard Shadow Experiment Analysis of 3D Image Engine

(1) Experimental analysis of HDR module special effects

In this paper, the HDR effect module is tested on the PC based on the Windows system. Figure 6 is the frame ratio comparison before and after the HDR is applied in the experiment.

As can be seen, in Figure 6 (a), 133.76FPS has the best rendered frame rate when the HDR effect is turned on. In Figure 6(b), the best frame rate when the HDR effect is 233.44FPS. Hence, when the HDR effect is turned on, the rendering frame rate is reduced, but the display effect is enhanced. HDR is more detailed and shows some previously blurred areas.

(2) Experimental analysis and comparison of soft and hard shadows

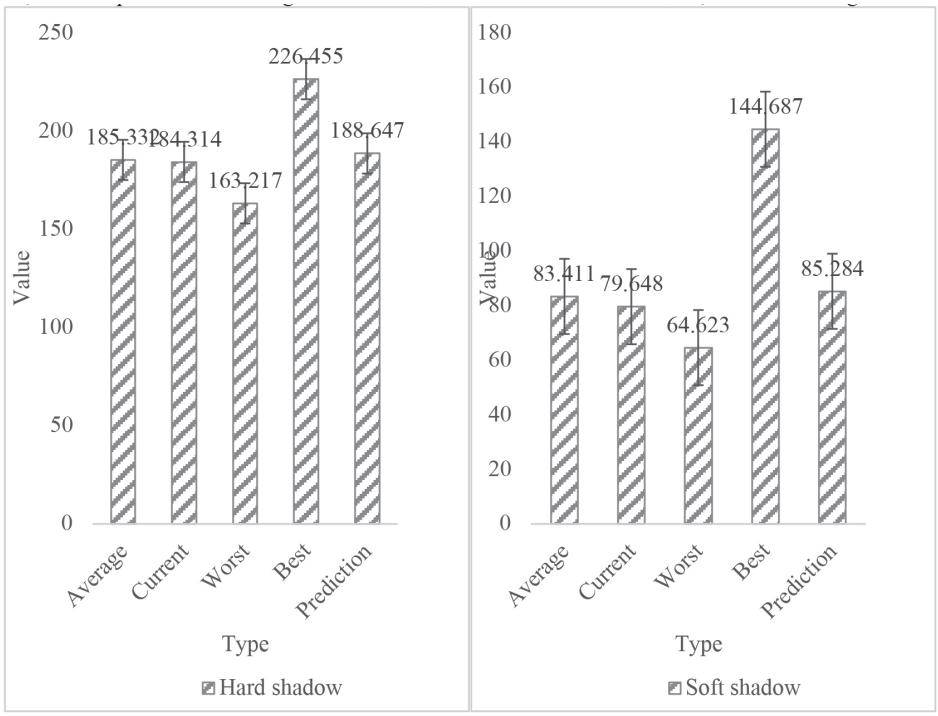
Where there are light and obstacles, there are shadows. The shadow also inidcates the relationship between the object and the light source, making the virtual computer world more real. High-quality shadow processing is also essential for some photo-level graphics rendering or light simulation. At the same time, shadows can help us to better understand the goals in the scene. Based on this, a new method based on softening shadows is proposed and applied to [17] in the animation of 3D graphics engine.

Because the light is blocked and cannot penetrate, a deeper area of light will appear on the surface of the object itself and other objects; this is known as "shadow". The shadow is divided into a self-shadow area and a projection shadow area. The self-shadow area is the dark area of the object itself, while the projection shadow area is blocked by the object; hence, it cannot illuminate other objects. Generally speaking, under light conditions, the shading condition of the target can be divided into two types: the occlusion body and the receiver body.

1) Soft shadows and hard shadows

Hard shadow occurs when a spot in a scene can exist in only two situations, namely, internal shadow and external shadow. Since the size of the light source is fixed, in the case of a penumbra, there is no hard shadow.

Soft shadow refers to a scene with both the original shadow area and the penumbral area. Compared to the hard shadow, the soft shadow is more suitable for the light sources with a specific size, such as the surface light source or the body light source. This is the characteristic of light sources in the real world, so soft shadows can make the scene more realistic.



(a) Hard shadow (B) Soft shadow

Figure 7 Rendering comparison of hard shadows and soft shadows.

2) Performance analysis

In this paper, the rendering effect of the 3D image engine animation system is tested on a Windows-based computer, and compares the rendering frame rate of hard shadow and soft shadow, as shown in Figure 7.

Figure 7 (a) shows that the best rendering frame rate with hard shadows is 226.455FPS and the worst-case value is 163.217FPS. With soft shadows found in Figure (b), the best rendering frame rate is 144.687FPS and the worst is 64.623FPS. This indicates that the soft shadow algorithm can meet the requirements of both reality and real-time performance.

4. CONCLUSIONS

This paper presents the design and implementation of the animation system of 3D graphics engine, and discusses the core technology. First, this paper introduces the research background of this topic, and analyzes and prediction, and proposed specific design scheme. We study the difficulties encountered in the animation design of the engine, and combine the relevant results of the current excellent 3D graphics engine. This paper improves the performance and methods by applying an in-depth understanding of the existing algorithms. Compared with previous algorithms, this method is more efficient and easy to implement. Finally, we demonstrate the effectiveness and performance of the proposed method through experiments. This study has achieved the expected results, but there are still some areas requiring improvement, specifically the design and implementation of the 3D graphics engine. Although the current system can meet the requirements of drawing, in terms of ensuring drawing accuracy, this paper does not address the animation system of 3D graphics engine. Hence, the application is limited and requires further improvement.

REFERENCES

1. Xi Chen, and Yongbin Tang. Data Center Energy Management Based on Cloud Computing and Artificial Intelligence. International Journal of Engineering Intelligent Systems, 32.3 (2024): 257–266.
2. Hiranyachattada Tiantada, and Kampanat Kusirirat. “Using Mobile Augmented Reality to Enhancing Students’ Conceptual Understanding of Physically-Based Rendering in 3D Animation.” European Journal of Science and Mathematics Education, 8.1 (2020): 1–5.
3. Zhang Lei. “Application research of automatic generation technology for 3D animation based on UE4 engine in marine animation.” Journal of Coastal Research, 93.SI (2019): 652–658.
4. Lamberti Fabrizio, Alberto Cannavo, and Paolo Montuschi. “Is immersive virtual reality the ultimate interface for 3D animators?” Computer, 53.4 (2020): 36–45.
5. Paier Wolfgang, Anna Hilsmann, and Peter Eisert. “Interactive facial animation with deep neural networks.” IET Computer Vision, 14.6 (2020): 359–369.
6. Hoon, Loh Ngiik, and S.S. Shaharuddin. “Learning effectiveness of 3D hologram animation on primary school learners.” Journal of Visual Art and Design, 11.2 (2019): 93–104.
7. Kaushal Rajesh Kumar, and Surya Narayan Panda. “A Meta Analysis on Effective Conditions to Offer Animation Based Teaching Style.” Malaysian Journal of Learning and Instruction, 16.1 (2019): 129–153.

8. Abdrashitov Rinat, Alec Jacobson, and Karan Singh. “A system for efficient 3D printed stop-motion face animation.” ACM Transactions on Graphics (TOG), 39.1 (2019): 1–11.
9. Alenezi, A. (2024). Online Surveillance of IoT Agents in Smart Cities Using Deep Reinforcement Learning. International Journal of Intelligent Information Technologies (IJIIT), 20(1), 1–15.
10. Weibin Wang. Parallel Computing Technology Based on Computer Simulation. International Journal of Engineering Intelligent Systems, 32.3 (2024): 245–255.
11. Caushaj, E., Sugumaran, V. Classification and security assessment of android apps. Discov Internet Things 3, 15 (2023).
12. Ji, Yuanyuan. “Use of virtual reality technology in animation course teaching.” International Journal of Emerging Technologies in Learning (iJET), 16.17 (2021): 191–208.
13. Xu, Z., Jain, D.K., Neelakandan, S. et al. Hunger games search optimization with deep learning model for sustainable supply chain management. Discov Internet Things 3, 10 (2023).
14. Schulz, Trenton, Jim Torresen, and Jo Herstad. “Animation techniques in human-robot interaction user studies: A systematic literature review.” ACM Transactions on Human-Robot Interaction (THRI) 8.2 (2019): 1–22.
15. Hanif, Muhammad. “The Development and Effectiveness of Motion Graphic Animation Videos to Improve Primary School Students’ Sciences Learning Outcomes.” International Journal of Instruction, 13.3 (2020): 247–266.
16. Bao, Wenrui. “The application of intelligent algorithms in the animation design of 3D graphics engines.” International Journal of Gaming and Computer-Mediated Simulations (IJGCMS), 13.2 (2021): 26–37.
17. Samah, Siti Nur Shuhada Abu. “The Efficacy of Augmented Reality on Student Achievement and Perception among Teluk Intan Community Col lege Student in Learning 3D Animation.” International Journal of Multimedia and Recent Innovation, 2.2 (2020): 87–95.