WebGL-based virtual reality technology construction and optimization

Zhongming Zhang[a], Jian Xu[b], Xun Shen*[b], Hailing Zhao[a], Yaohui Niu[b]
[a]Power Exchange Center of Xinjiang Corporation, Urumqi 830063, Xinjiang, China; [b]School of Software, Nanjing University of Information Science and Technology, Nanjing 210044, Jiangsu, China

ABSTRACT

With the rapid development of mobile Internet and the increasing maturity of virtual reality technology, there is an increasing demand for immersive virtual reality experiences on Web platforms. Therefore, this paper aims to explore the development of virtual reality technology on the Web side, and WebGL technology is adopted to realize this goal. The paper details several commonly used WebGL development frameworks and evaluates them in various aspects. It also proposes a generic framework design for Web3D modeling and a specific construction method for realizing 3D virtual scenes. Finally, the paper discusses the performance optimization techniques for Web3D from the perspectives of modeling, loading, rendering, and memory optimization. In conclusion, this paper explores WebGL-based virtual reality on web platforms, presenting a general Web3D modeling framework and performance optimization techniques for efficient and responsive 3D virtual scenes.

Keywords: WebGL, virtual reality, framework design, 3D modeling approach, performance optimization

1. INTRODUCTION

Currently undergoing a revolutionary change with the rise of the meta-universe and the concept of Web 3.0, Web3D technology continues to grow, with its applications expanding from gaming and e-commerce to educational platforms and a variety of virtual events. Emerging technologies such as Web Extended Reality enable the development of immersive VR and AR experiences directly in the web browser[1]. The Web, as a lightweight cross-platform mainstream browsing environment, has a huge user base in the mobile Internet era. With the emergence of various 3D graphics APIs such as X3D, WebGL, and WebGPU, the Web side has become the core technology for creating 3D applications. Highly realistic and immersive Web3D applications are beginning to be presented to users[2,3].

This paper focuses on the use of WebGL technology to realize the application of virtual reality technology on the Web side. The development of WebGL is led by the Khronos team, which is a consortium of industry-leading hardware and software vendors. Currently, major browser vendors, such as Google (Chrome), Mozilla (Firefox), Opera (Opera), and Apple (Safari), are supporters and active participants of WebGL[4].

WebGL is a free cross-platform 3D API based on OpenGL ES 2.0 that allows hardware-accelerated 3D graphics rendering in Web browsers using JavaScript and OpenGL ES 2.0. WebGL provides powerful 3D graphics capabilities for Web applications with powerful 3D graphics capabilities[5]. The generic system architecture diagram of WebGL is shown in Figure 1. It uses the JavaScript language for rendering and displaying 3D graphics in HTML5 Canvas tags and accesses different graphics drivers through the OpenGL ES API. In addition, it utilizes GPUs to achieve acceleration of 3D rendering at the hardware level.

*1632824394@qq.com

International Conference on Optics, Electronics, and Communication Engineering (OECE 2024), edited by Yang Yue, Proc. of SPIE Vol. 13395, 133951L · © 2024 SPIE · 0277-786X · Published under a Creative Commons Attribution CC-BY 3.0 License · doi: 10.1117/12.3048388

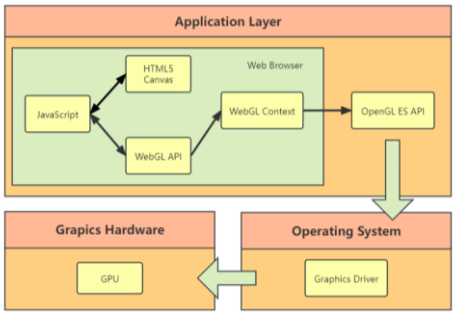

Figure 1. System architecture based on WebGL.

WebGL has a wide range of application areas and advantages in Web applications. First, it provides developers with the ability to create high-quality, realistic virtual reality experiences in Web browsers. By utilizing hardware-accelerated 3D graphics rendering, WebGL can render complex 3D scenes, models, and effects, thus providing users with an immersive interactive experience. Second, WebGL is fully integrated with HTML5 and other Web technologies and can seamlessly interact with DOM interfaces and other Web APIs. This allows developers to leverage existing Web development skills and tools to build WebGL applications, as well as easily integrate WebGL with other Web features such as data visualization, audio, and video[6]. In addition, WebGL is cross-platform and open-standard, running on a variety of devices and operating systems, including desktop computers, mobile devices, and virtual reality devices. This allows developers to create unified virtual reality experiences for different platforms and devices, thus providing a wider reach and user base.

Therefore, based on the above advantages, the WebGL API, which is widely used nowadays, was chosen for the construction. In this paper, several commonly used WebGL development frameworks are first introduced in detail and evaluated in various aspects. Then, a framework design based on Web3D modeling is proposed, which includes a scene management module, a model loading module, and a user interaction module, as well as a specific construction method for realizing 3D virtual scenes. Finally, performance optimization techniques for Web3D are discussed in detail, including methods and strategies for model optimization, loading optimization, rendering optimization, and memory optimization.

2. WEBGL DEVELOPMENT FRAMEWORK

Based on the development of Web3D technology, a variety of WebGL-based development engines have emerged (as shown in Table 1). Creating a 3D scene in the browser by directly invoking hardware acceleration for graphics rendering is a rather cumbersome task, so choosing a suitable development framework is crucial.

Table 1. Web3D engines.

Engine	Company	Release year	Supported languages	Applications
Three.js	Ricardo Cabello	2010	JavaScript	WebGL-based 3D visualizations JavaScript library
Babylon.js	Microsoft	2013	JavaScript/TypeScript	Feature-rich framework for 3D game development
PlayCanvas	Snap Inc.	2011	JavaScript	Cloud-based platform for creating interactive 3D games
A-Frame	MozVR	2015	JavaScript	Open-source VR framework using HTML markup
Filament	Google	2018	JavaScript/Java/C++	Real-time rendering engine for high-quality graphics

Three.js is based on the WebGL package of an easy-to-use and lightweight 3D library, it uses JavaScript to write, and the class belongs to WebGL. Three.js WebGL provides a very good encapsulation of the interface, simplifying many details, greatly reducing the learning cost, and greatly improving performance, functionality is also very powerful. three.js can be used to better solve the problems encountered in the development process and can make full use of the renderer, model, light source, and so on for the creation of 3D scenes[7]. The use of Three.js can better solve the problems encountered in the development process and can give full play to the renderer, model, light source, etc. for the creation of 3D scenes. Because Three.js is written in JavaScript, it makes Three.js more compatible, which enables secondary development and

hotspot interactions, leading to greater interactivity when users use Web3D. At the same time, Three.js has a wide range of 3D rendering features, including geometry, materials, lighting, and animation, which are highly customizable and can be used to build complex 3D scenes. It provides detailed documentation and an active community to help developers get started and solve problems quickly. In addition, Three.js is highly affordable as an open-source and free-to-use library, making it very cost-effective for developers and organizations. As a stable library with regular updates and improvements, Three.js has been widely adopted in the industry and boasts a long presence since 2010, as well as a large user base and a wealth of resources. This allows developers to benefit from a wide range of communities and resources to better support their development efforts.

Babylon.js is a Web3D game engine released by Microsoft in 2013, with an underlying layer based entirely on the WebGL graphics API. The engine is written in JavaScript with support for the TypeScript language. With a clean API, extensive documentation, and a list of tutorials, Babylon.js can be used to build interactive 3D presentations, 3D Web-ready product demos, games, VR applications, and complex architectural simulations. Its focus is in the area of Web3D game development. Babylon.js, a powerful 3D engine for web applications that provides rendering, physics, animation, and VR or AR features, has a reputation for its excellent expressiveness and comprehensive support for 3D and 2D graphics[8]. It is known for its ease of use and is suitable for developers with JavaScript skills and some knowledge of 3D graphics. The engine has comprehensive and detailed documentation and an active community, making it easier for developers to utilize its features. On the economic side, Babylon.js is an open-source engine and free to use, making it an affordable option for 3D web development. In addition, Babylon.js is a stable and well-maintained library that is constantly releasing updates and enhancements to ensure its stability and reliability. Since its introduction in 2013, Babylon.js has gained wide acceptance in the industry and is considered very mature for building complex 3D web applications.

PlayCanvas is an open-source Web3D engine developed by Will Eastcott et al. It includes a 3D game engine, an interactive 3D application engine, and a proprietary cloud-hosted creation platform. The engine allows for simultaneous editing from multiple computers through a browser-based interface and was acquired by Snap Inc. in 2017. With an impressive feature list and a specialized cloud-hosted platform, PlayCanvas provides game developers with the ability to build web-first graphics to enrich game content. It can also be used to build augmented reality (AR) and virtual reality (VR) applications. PlayCanvas is a 3D game engine designed for web-based development, offering a visual editor, physics effects, scripting, and multiple export options. It is suitable for use by technicians of all levels, with the visual editor making it more beginner-friendly. On the economic side, PlayCanvas offers both free and paid plans at reasonable prices suitable for small and medium-sized projects. Moreover, PlayCanvas serves as a stable platform with regular updates and improvements to ensure its stability and reliability. Although PlayCanvas is not as well-known as Unity, it has been around for several years and the community is growing. PlayCanvas has built a reputation in the industry, maturing and becoming suitable for building a variety of complex 3D web applications.

A-Frame is a framework for building virtual reality (VR) and augmented reality (AR) experiences on the Web. It is HTML-based and built on top of WebXR and Three.js with a powerful entity framework at its core. A-Frame is widely used by the world's top companies including Disney, Google, Mozilla, NASA, Samsung, Sony, and Toyota. As an open-source framework, A-Frame provides an extensible and reusable architecture that gives developers the freedom to use technologies such as JavaScript, DOM API, Three.js, WebXR, and WebGL. It simplifies the process of creating 3D and VR content by using an HTML-like markup language and a component-based system. A-Frame was designed with the goal of being easy to get started with for web developers, even those with limited experience in 3D programming. It uses an HTML-like syntax, making it easier for developers to understand and use. As a free open-source framework, A-Frame is economical and well-suited for web-based VR and AR development. And A-Frame is actively maintained and has a stable developer community. It is built on a stable underlying technology, so it has stability and reliability. Although A-Frame is relatively new compared to some more mature technologies, it has already gained widespread attention in the industry and is considered a mature framework for a variety of Web-based VR and AR applications. It has a wide range of applications to meet the needs of different projects and continues to attract the attention of more and more developers and companies.

Filament is a cross-platform real-time rendering engine developed and maintained by Google, focusing on mobile devices and desktop platforms. As a physically-based real-time renderer, Filament aims to provide advanced rendering techniques such as Physically Based Rendering (PBR), global lighting, shadows, and realistic materials. It is an open-source WebGL real-time 3D rendering engine written in C++ and is designed to be a mobile-first solution. Filament has broad platform support including Android, iOS, Linux, macOS, and Windows. Its main goal is to build high-performance 3D rendering applications for the Web. Filament supports a wide range of graphics APIs, including WebGL 2.0, Vulkan, Metal, OpenGL 4.1, and OpenGL ES 3.0[9]. Filament is packed with powerful features and flexible

customizability. It provides an intuitive and easy-to-use API and detailed documentation to enable developers to get started and develop quickly. In addition, Filament supports multiple programming languages, such as C++ and Java, providing developers with more choices. As a free and open-source engine, Filament provides an affordable rendering solution. It has good stability and performance, which has been extensively tested and optimized. Thanks to Google's support and maintenance, developers can expect continuous updates and technical support. Although Filament is relatively new and has not yet reached the level of maturity of some of its competitors, it is widely recognized for its features and performance. Over time, Filament will continue to evolve and improve to become a competitive real-time rendering engine of choice.

3. WEB3D MODELING FRAMEWORK DESIGN

In order to facilitate the application of WebGL technology to real-world scenarios more conveniently, an efficient Web3D modeling framework is designed in this paper. The overall structure of the framework is shown in Figure 2, which aims to provide better reusability, extensibility, flexibility, and maintainability to better meet the development needs of various Web applications.

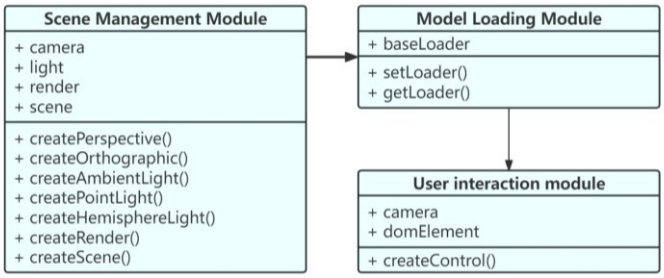

Figure 2. Web3D modeling framework.

3.1 Scene management module

When the framework starts, the first thing to run is the scene management module, which provides the basic components for the application scene rendering, including cameras, lights, renderers, etc. These components are essential for rendering an application scene[10].

Camera components are elements used to observe the model from a point of view and used in rendering the scene. Depending on the needs of the scene, different visual effects can be achieved by choosing observation lenses with different characteristics. For example, an orthogonal lens ensures that the geometry is not distorted by changes in viewing distance and is typically used in engineering modeling and mathematical geometry. The perspective lens, on the other hand, simulates the observation effect of the human eye, so that the visual effect of the scene will be similar to the deformation effect observed by the human eye with the field of view and observation distance of the lens. In addition, the camera component can initialize various parameters of the camera according to specific needs, including information such as angle, viewing distance, and aspect ratio. If no specific parameters are passed in, the system will use the default parameter values for initialization. This design provides more flexible configuration options to obtain the desired visual effect.

As an important component of the scene, the light plays a crucial role in the rendering effect of the 3D scene. Lack of light will make the whole scene in darkness, which seriously affects the rendering results. When choosing a light source, you can select it according to the specific scene and model requirements. For example, a basic ambient light source provides a uniform lighting environment in the scene so that the entire scene is properly illuminated. A parallel light source simulates a parallel light source from an infinite distance and is often used to simulate sunlight. A point light source simulates the effect of a single point light source, and the intensity and range of its light depends on the angle of radiation of the point light source and the intensity of the light and other attributes. The properties of the light source include color, intensity, etc. These properties can be adjusted according to specific needs to achieve the desired visual effect.

The renderer component determines on which element of the page the rendering result should be drawn and in which way. The WebGL renderer used in this article takes advantage of GPU hardware acceleration to improve rendering performance, resulting in smoother rendering results. The renderer component consists of a set of properties such as the width, height, background transparency, and background color of the renderer. By adjusting these properties, we can configure the renderer to meet specific needs. For example, setting the width and height of the renderer ensures that the

rendered result matches the size of the target element. Adjusting the background transparency and color properties can achieve different background effects and provide a suitable background visual presentation for the scene rendering result.

3.2 Model loading module

The model loading module is responsible for loading and rendering models in the application scene. The module has support for different model formats and provides model loading, position adjustment, and animation control to display and manipulate models in the scene.

Unlike traditional 3D models in C/S mode, models in Web3D need to be parsed by the browser and image processed by the GPU, and then the result is returned to the browser for rendering and display. Model files can be loaded using loaders provided by Web3D frameworks such as Three.js. These loaders support various formats such as OBJ, FBX, GLTF, etc. to load model files into the WebGL rendering environment[11]. Besides, the Web3D framework also provides corresponding interfaces or methods to modify the transformation properties of the model to adjust its position, rotation scaling, etc. It also provides animation systems or libraries to control the animation behavior of the model. By defining keyframes and animation sequences, the dynamic effects of the model can be realized.

Considering that different application scenarios may require loading models in different formats, the design can be done using the strategy pattern. The abstract strategy class, which contains the methods that the loading algorithm must have, is defined first. Specific loading algorithm strategy classes are then created, each of which also implements the methods in the abstract strategy class for loading 3D models of the corresponding format. Through the design of the strategy pattern, the encapsulation of the loading algorithms of different formats is realized, so that when facing models of different formats, only the loading strategy class of the corresponding format needs to be passed in to realize loading. The advantage of this design is that it improves the reuse rate of the code, avoids the writing of redundant loading functions, and also increases the flexibility of the code so that the loading algorithms can be quickly switched according to the needs of the scene.

3.3 User interaction module

A user interaction module to handle user interaction with the 3D scene. This module can include functions such as controllers, hotspot events, and user input processing. It can provide different control modes, such as mouse control, keyboard control, etc., to meet the user's needs when viewing and manipulating the models in the scene[12].

The core component of the user interaction module is the controller. The controller can be used in both mouse control and keyboard control modes. For the mouse control mode, you can use the mouse event listener provided in the Web3D development framework to capture the mouse movement, click, and scroll wheel events. By listening to these events, it is possible to realize the operations such as panning, rotating, and zooming of the scene. Users can zoom in and out of the Web3D scene by using the mouse wheel or other gestures to better view the models and details in the scene, and move the whole Web3D scene by dragging and dropping it with the mouse or keyboard control to view the content in the scene from different angles. For the keyboard control mode, you can also use the keyboard event listener in the framework to capture the events of keyboard keys. By defining the mapping of keyboard control, you can use the arrow keys to move around the scene and use specific keys to trigger specific interactions.

In addition, the hotspot event function is also relatively important in the user interaction module. By adding the interaction hotspot interface, when the user hovers or clicks the mouse on the hotspot, it will trigger the corresponding event handler function. These events can be used to display more information about the model, show pop-up boxes, trigger animation effects, and so on.

4. CONSTRUCTION OF 3D VIRTUAL SCENES

After building the basic Web3D fundamental framework, the construction of 3D virtual scenes can be started. This paper proposes the following steps, i.e., data collection, 3D modeling, website development, and testing[13].

Data collection is needed first. The content of data collection includes the shape, size, and texture of the objects in the scene. Through on-site survey, image acquisition, or other data acquisition methods, the accurate information of each object in the scene is collected to provide basic data for subsequent modeling.

After completing the data collection phase, 3D modeling can begin[14]. According to the research needs and development requirements, choose a development engine suitable for WebGL. Then start from the basic geometric model, gradually

add details, and apply textures and materials to enhance the visual effect of the model.

In addition, considering the characteristics of Web-side development, it was necessary to create a virtual reality Website that converts 3D models built in the form of virtual reality into models that can be displayed using a Web browser. In order to provide a richer user experience, user interaction features need to be added. For example, by using the mouse or keyboard, users are allowed to control the viewpoint or interact with the model, which increases the user's sense of involvement and exploration.

Finally, a testing step is essential to ensure that the constructed 3D virtual scenes are displayed properly on the web page and have good loading performance[15]. Testing includes verifying the correctness and stability of the model on the web page and evaluating the loading time of different scenes on the website. Through testing, potential problems can be identified and resolved to optimize the rendering performance and user experience of the scenes.

5. WEB3D PERFORMANCE OPTIMIZATION TECHNIQUES

After building a basic 3D virtual scene on the website, it is crucial to optimize Web3D performance to keep the application smooth and responsive.

5.1 Model optimization

Model and texture data are the most voluminous data during the loading process of a Web3D application, so optimizing for this data is critical to improving performance[16].

At the modeling stage, it is necessary to develop appropriate specifications. Under the premise of ensuring the appearance effect, a streamlined model should be used as much as possible to reduce the number of faces in the model. In addition, when selecting the constituent nodes of the model, priority should be given to regular geometric nodes and the number of vertices of complex nodes should be minimized. After the model is completed, selecting applicable model transfer formats, such as FBX and GLTF, can further reduce the size of the model file and improve the loading efficiency.

In order to reduce the size of the mapping file, appropriate measures should be taken. First of all, the appropriate size of the texture maps should be selected according to the need. Generally, it is not recommended to exceed 4K, and it is better to keep it at 1024 or 2048. It is better to use the size of 2n in order to get a better performance in the texture sampling in the graphic processing unit. It's also a good idea to look at whether textures can be merged and reused to further reduce the number of textures and file sizes. Binomial and Google's Basis Universal technology, which was launched in May 2019, supports a wide range of commonly used compressed texture formats, converting PNGs to Basis files to achieve file sizes that are similar to JPEGs, but on a smaller GPU. The use of Basis Universal technology maintains GPU performance efficiency while improving image transfer performance on the web side.

5.2 Load optimization

3D models can take a long time to load on a web page due to network speed limitations, so the loading time needs to be minimized to improve the user experience.

For large scenes, especially BIM or GIS scenes, loading all models at once may result in a long waiting time for users. Therefore, a split-packet streaming loading method can be used to split large data into multiple small packets, load a part of the model in each packet, and transfer it to the rendering main thread for rendering after loading and parsing are completed. This can effectively improve the loading speed and rendering efficiency of the scene[17].

Model grid data compression can be done using gzip compression, which can lead to better compression ratios for some purely character-encoded models such as obj, dae etc.[18]. It is worth noting that gzip compression is an operation that is done by default between the server and the browser without additional developer intervention and is therefore imperceptible to the developer. Therefore, gzip compression is recommended for files that use this type of model. This way files that have been gzip compressed will also have a much smaller amount of data in transit.

5.3 Rendering optimization

Using efficient rendering techniques, you can speed up rendering, improve system responsiveness, and render more realistic and satisfying graphical results.

To achieve realistic results, designers often use complex geometric models, consisting of a large number of polygons. However, complex geometric models can greatly increase the amount of browser computation, resulting in a slow scene

to navigate[19]. To solve this problem, models with different resolutions can be created and applied depending on the distance of the observer. There are three general types of detail models: low-resolution models are used when the observer is at a distance, preserving basic features; medium-resolution models are switched to as the observer gets closer, replacing complex details with simplified geometry; and high-resolution models are used when the observer is up close, presenting all the details.

In addition, culling minimizes the amount of data submitted to the GPU to reduce the rendering burden. Common culling methods include apparent vertebrae culling, backside culling, and occlusion culling[20]. Visual vertebrae culling determines whether to render by determining whether an object lies within the visual vertebrae. This has the advantage of avoiding rendering invisible objects, thus reducing unnecessary computation and rendering overhead. Backside culling, on the other hand, excludes invisible backsides by determining whether to render based on the front and back sides of the object[21]. Occlusion culling, on the other hand, determines in advance the objects that do not need to be rendered based on their occlusion relationship after camera culling to reduce GPU rendering pressure.

5.4 Memory optimization

For some large Web3D application scenarios, fine control of memory is especially important.

A common optimization method is to use a particle system instead of a large number of individual particles[22]. When a large number of particles need to be rendered, the particle system combines them as a whole, sharing a set of attributes and behaviors, rather than allocating separate memory for each particle. This technique significantly reduces memory consumption and excels in rendering performance.

In addition, freeing memory in a timely manner is a critical task. Instantly destroy and free the JavaScript memory occupied by an object when it is no longer in use. This can be done by dereferencing, invoking a garbage collection mechanism, or manually freeing resources[23]. For large data such as model data, it is also important to release it from memory in a timely manner. In addition to releasing memory in a timely manner, users are advised to use browsers such as 64-bit Chrome or Firefox to fully utilize the larger JavaScript heap memory to avoid the risk of JavaScript heap overflow.

6. CONCLUSION

Web3D technology, as a practical, pioneering, and comprehensive technology, is increasingly occupying an important position in the development of the Internet and is expected to gradually replace the mainstream 2D graphics technology, bringing potential with revolutionary changes for future Internet applications. In this paper, systematic research on the construction and optimization of WebGL-based virtual reality technology on the Web platform is carried out, which proposes the framework design of Web3D modeling and the specific construction method for realizing 3D virtual scenes, and conducts an in-depth discussion on performance optimization techniques, aiming to provide inspiration for related practitioners.

Future research will continue to be dedicated to performance optimization aspects, in-depth research and optimization of algorithms, and rendering technology, focusing on improving user interaction, so that users can interact with the virtual environment in a more natural and convenient way, thus enhancing the user's sense of participation and satisfaction. Through continuous technical improvements and innovative applications, we can expect the authenticity, immersion, and diversity of virtual reality experience on the Web side to be further enhanced, which will in turn promote the application and development of virtual reality technology in various fields and bring positive impacts to people's life and work.

ACKNOWLEDGEMENT

This work was supported by the Headquarters Management Science and Technology Project of State Grid Corporation of China under Project No. 5108-202355441A-3-2-ZN.

REFERENCES

[1] Gesquière, G. and Manin, A., "3D visualization of urban data based on CityGML with WebGL," International Journal of 3-D Information Modeling, 1-15 (2012).

[2] Jiang, H., “The construction of virtual simulation platform for Pingtan experimental area based on HTML5 and WebGL,” Enterprise Information Systems, 14(1), 1-18 (2019).
[3] Lu, Z., Guo, W., Zhao, S., et al., “A cross-platform Web3D monitoring system of the three-machine equipment in a fully mechanized coalface based on the skeleton model and sensor data,” Journal of Sensors, 1-14 (2020). http://dx.doi.org/10.1155/2020/3147352. DOI:10.1155/2020/3147352.
[4] Sun, F., Zhang, Z., Liao, D., et al., “A lightweight and cross-platform Web3D system for casting process based on virtual reality technology using WebGL,” The International Journal of Advanced Manufacturing Technology, 80, 801-816 (2015).
[5] Chaturvedi, K., [Web-Based 3D Analysis and Visualization Using HTML5 and WebGL], University of Twente, (2014).
[6] Peng, C., Chen, L., et al., “Construction of a theme virtual learning community based on Web3D,” 2010 International Conference on Networking and Digital Society, Wenzhou, China, (2010). http://dx.doi.org/10.1109/icnds.2010.5479255.
[7] Liu, C., Huo, Y., Zhang, Y., et al., “A review of real-time rendering technology based on mobile internet platforms,” Journal of Image and Graphics, 27(06), 1877-1897 (2022).
[8] Yu, G., Liu, C., Fang, T., et al., “A survey of real-time rendering on Web3D application,” Virtual Reality & Intelligent Hardware, 2023, 5(5): 379-394.
[9] Xu, X., “An analysis of several typical Web 3D development technologies,” 2012 International Conference on Computer Science and Information Processing (CSIP), Xian, Shaanxi, China, (2012). http://dx.doi.org/10.1109/csip.2012.6309062. DOI:10.1109/csip.2012.6309062.
[10] Zhou, H., “Research and application of Web3D visualization technology,” Wisdom China, 78-79(2022).
[11] Cheng, L., [Design and Implementation of Reusable IoT Web3D Framework], Hangzhou: Hangzhou Normal University, Master’s Thesis, (2020).
[12] Bian, M. J., Gao, H. H., et al., "Research and application of Web3D exhibition based on WebGL and HTML5,” Proceedings of the 2015 International Conference on Electrical, Automation and Mechanical Engineering, Advances in Engineering Research, Phuket, Thailand, (2015).
[13] Wardijono, B. A., Wardhani, I. P., Chandra, Y. I., et al., “3D virtual environment of Taman Mini Indonesia Indah in a web,” Journal of Physics: Conference Series, 012158 (2018).
[14] Vyas, Y., Campbell, E., Anderson, S., et al., “A workflow for Web3D interactive outdoor scene visualization,” In: Proceedings of the 22nd International Conference on 3D Web Technology, (2017).
[15] Scianna, A., La Guardia, M. and Scaduto, M. L., “Sharing on web 3D models of ancient theatres. a methodological workflow,” The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 483-490 (2016).
[16] Evangelidis, K., Papadopoulos, T., et al. “3D geospatial visualizations: Animation and motion effects on spatial objects,” Computers & Geosciences, 200-212 (2018).
[17] Li, L., Qiao, X., Lu, Q., et al., "Rendering optimization for mobile web 3d based on animation data separation and on-demand loading,” IEEE Access, 8, 88474-88486 (2020).
[18] Liu, C., Ooi, W. T., Jia, J., et al., "Cloud baking: collaborative scene illumination for dynamic Web3D scenes,” ACM Transactions on Multimedia Computing, Communications, and Applications, 1-20 (2018).
[19] Peng, C., “The research and design of 3D Web guide system based on WebGL,” The 26th Chinese Control and Decision Conference (2014 CCDC), 4052-4054 (2014).
[20] Hulusic, V., “Optimization threshold in sarajevo city hall virtual model for efficient web presentation,” In: Central European Seminar on Computer Graphics, Slovakia 2007. 2007.
[21] Yan, Y., Ma, J., Zhang, Y., et al., “Research on real-time model optimization methods for VRML in Web3D,” Ship Electronic Engineering, 30(05), 135-138 (2010).
[22] Zhou, W., Tang, K. and Jia, J., “S-LPM: segmentation augmented light-weighting and progressive meshing for the interactive visualization of large man-made Web3D models,” World Wide Web, 1425-1448 (2018).
[23] Toasa, R. M., Baldeón Egas, P. F., et al., “Performance evaluation of WebGL and WebVR apps in VR environments,” Advances in Visual Computing, Lecture Notes in Computer Science, 564-575 (2019).