Lightweight Application Of Signal BIM Models Based On GlTF And WebGL

Yunshui Zheng* and Yingying Ding

School of Automation & Electrical Engineering, Lanzhou Jiaotong University, Lanzhou 730070, China

Corresponding author. E-mail: 2433782572@qq.com

Received: May 09, 2022; Accepted: Sep. 15, 2022

Without affecting the application effect of BIM (Building Information Modeling) model, appropriate simplification of the model is a key step to promote BIM technology in the construction of engineering informatization. Aiming at the problems of large amount of model data, weak universality of data format and low network transmission efficiency, a research method for lightweight visualization application of signal BIM models based on glTF format and WebGL technology is presented. Firstly, the geometric spatial information and semantic attribute information required by the models are extracted through the idea of digital - analog separation, so as to realize the lightweight of the internal data of the models. Secondly, the QEM (Quadric Error Metrics) algorithm is used to simplify the external geometric triangles of the models and the Draco algorithm is used to compress and reduce the model file. On this basis, the geometric information and attribute information of the models are stored separately in the glTF file and MySQL database. Finally, the lightweight processed files are connected to the developed Three.js rendering engine to realize the development of visualization application based on WebGL technology. The research shows that this method realizes the data format conversion of the signal BIM models, and can effectively reduce the data volume of the model at the same time. In addition, the lightweight display and attribute information query of the models on the web pages are realized. It is beneficial to improve the application value of the signal BIM models in the digital development of railway signals.

Keywords: railway signal; BIM lightweight; glTF format; WebGL technology; visualization application

http://dx.doi.org/10.6180/jase.202308_26(8).0003

1. Introduction

With the advantages of visualization, coordination, simulation and integration, BIM technology has been rapidly applied and developed in the engineering industry at home and abroad [1]. As the advantages of this design concept become more and more obvious in the project field, it has become an inevitable trend to use BIM technology to promote the construction of digital railway [2]. Railway signal projects have the characteristics of fine construction requirements, involving a wide variety and large number of equipment. And there is a phenomenon that non-designers do not use BIM software. As a result, in the process of construction and application of railway signal projects based on BIM, there are generally problems such as strong modeling expertise, large amount of model data, and high requirements for computer performance. And the direct use of the created BIM models will easily lead to problems such as long transmission time, poor browsing, and poor user experience. There may even be a terminal stuck phenomenon [3]. Ren et al. [4] realized the lightweight display of the models on the web browsers by using the detail slice suppression method, but the scheme resulted in the loss of detail features of the model. Liu et al. [5] determined that the glTF format has the advantages of better compression effect through comparative analysis, and realized the

lightweight of the file format based on glTF. However, it lacks the lightweight of the geometric appearance of the models, and there is insufficient in lightweight degree. Jin and Li [6] improved the transmission rate and rendering efficiency of the models by using four methods such as filtering geometric information and hierarchical loading. However, this scheme focuses on rendering optimization, and there are shortcomings in the lightweight of the model itself. Wang et al. [7] proposed a lightweight solution based on JSON file format and FastdFS file server, which made the models out of the constraints of IFC standard files and realized the Web-side display of the models. However, this method only lightens the surface of the models, without considering the non-geometric information processing of the models. It can be seen that the key problem to realize the application of signal BIM models on the web browser is the lightweight processing of the models. Although some scholars can realize the three-dimensional display function of related models, it is not enough to fully support the requirements for lightweight display of BIM-based railway signal projects applications.

In view of the above problems, this paper presents a visualization method of signal BIM models based on glTF format and WebGL technology. This method focuses on a lightweight web-based visualization system combining Revit and Three.js. Finally, three signal BIM models with different complexity are used as examples to discuss the lightweight application effect of signal BIM models based on glTF format and WebGL technology.

2. Research framework

The research framework of this paper is shown in Fig. 1, which is mainly composed of three modules: model lightweight processing, separate storage and visual application.

In the field of BIM, Revit software is the most widely used. It presents the idea of ' family', which makes the software have the functions of collaboration and component managemen [8]. Revit software is not only powerful in itself, but also provides application program interfaces compatible with multiple languages to meet more needs of users. Therefore, the RVT model file is selected as the data inputting. Firstly, on the one hand, the models are divided into geometric space information and semantic attribute information by digital-analog separation, so that unnecessary internal data information is eliminated once. On the other hand, the appearance triangles of the model are simplified by mathematical method. Secondly, the above processed data information is further compressed and lightened by using the Draco compression technology. Then

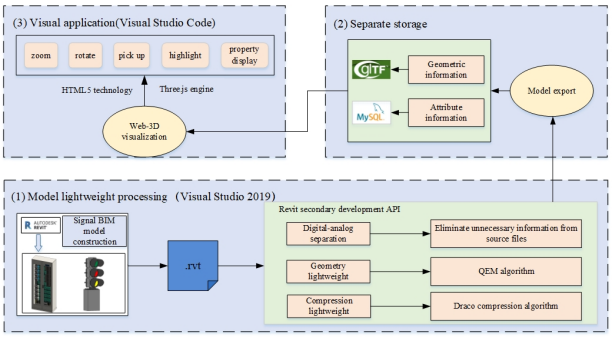


Fig. 1. The research framework

the model data is exported for separate storage. The model geometric space information is stored in the glTF file, and the semantic attribute information is stored in the MySQL database. Finally, the Web-3D visualization function module is developed based on HTML5 technology and Three.js framework. At the same time, functions such as zooming, rotating, highlighting and attribute information display are added to realize the interaction of users with the signal BIM models.

3. Lightweight processing of signal BIM models

The lightweight of the model means that under the premise of ensuring that the model is not distorted, the data volume of the model is reduced to the maximum extent through the algorithm, so that it can be displayed faster, lighter and more convenient. At the same time, the lightweight processing is also the only way for the model to realize multi-platform and cross-platform. The main elements necessary for model loading and rendering include vertex data, mesh data, normal vector and texture information [9]. However, in the Revit source file, in addition to the 'necessary' model elements, there are also reference elements such as axis net and view elements such as annotations. In order to eliminate these redundant information and avoid geometric information errors in the process of lightweight, this paper uses digital-analog separation to reduce the internal data of the models. In order to further reduce the data volume of the models, the geometric deletion method is used to simplify the triangular surface of the models to a certain extent. Then, Draco compression algorithm is used to compress files to speed up the loading rate of the models. According to the requirements of lightweight, this paper uses the mature and stable Autodesk Revit 2018 for modeling and loading tests, and uses Visual Studio 2019 and Revit2018 SDK work sets for development.

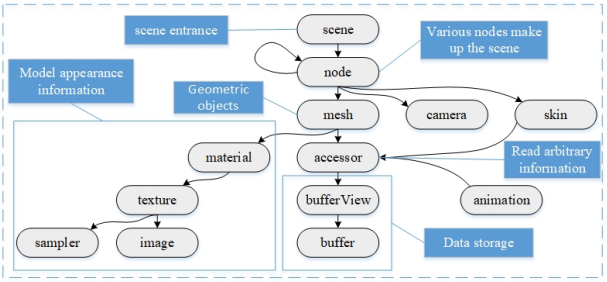


Fig. 2. The basic structure of the JSON file

3.1. Digital-analog separation

The data format of the original BIM model cannot be recognized and processed by the graphics engine, which hinders the application of the model in other platforms [10]. In order to make the BIM model no longer restricted by professional software, the choice of target file format is essential. For the selection of the target file format, this paper takes the distribution cabinet model created by Revit as an example to compare the application effects of glTF, OBJ and FBX data formats on the Three.js engine. The results are shown in Table 1. Considering the lightweight effect and loading effect, this paper selects glTF format as the target file format.

3.1.1. glTF file description

The data format of glTF file adopts the block-based storage [11]. The 3D scene description information such as nodes, cameras and rendering are stored in JSON text file. Texture information is stored as external texture file (.jpg / png). It is worth noting that a large number of model-related geometry, animation data and other information are stored in external binaries of high compression rate (.bin). When the Web side loads the glTF model, it just loads the JSON text file and access it by parsing its attribute fields [12]. The basic structure of JSON file is shown in Fig. 2.

3.1.2. The design of database

The database is developed based on MySQL and Navicat Premium. The database table established in this paper is mainly the attribute information table of the signal BIM models. This table mainly contains basic information such as the creation stage, name and rated voltage of the models. The connection between geometric information and non-geometric information is based on a unique ID.

3.1.3. Extraction of model data information

For the export of model geometry, Revit software provides a professional IExportContext interface. The interface includes 17 methods such as Start, OnViewBegin and OnElementBegin, which loop all the views, elements, entities,

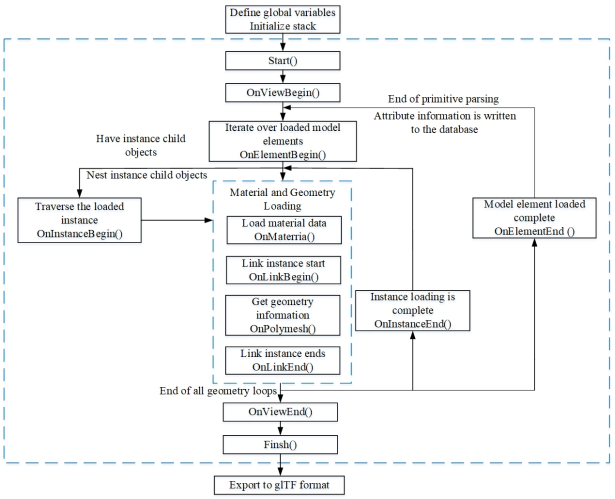


Fig. 3. Process of model data extraction

materials and other objects to obtain the geometric information of the model. Users can fill in the processing logic in the corresponding interface according to their own needs. Focusing on the goal of lightweight the internal data of the models in this paper, the geometric grid, material, texture and attribute information of the models are exported by custom IExportContext interface. The operation process is shown in Fig. 3.

The specific extraction process is as follows:

1. First global variables and stacks that store model objects are defined. Then a storage space is opened up by calling Start (), and the program obtains the material library address by reading the corresponding key value of the registry.

2. In OnViewBegin () , an operation is defined to initially reduce the surface of the model. OnElementBegin () and OnInstanceBegin () represent the start of export of element and family instance respectively. The Revit model is composed of multiple meshes or different instances, so when parsing, the instance sub-objects should be nested.

3. OnMaterial () obtains the corresponding material in the material library through the unique id (uuid) of the default material. If not obtained, a new material is constructed from the node and a connection is established. OnPolymesh () also uses uuid to convert the mesh object into the format of the target file. At the same time, when exporting the texture, vertices, UVs, and traction position are associated to ensure the corresponding relationship between geometric data and materials.

Table 1. The application effect of different data formats

Before conversion	Convert format	After conversion	Lightweight rate	Loading effect
6840KB	giTF	1687*KB*	75.3%	material, property
	OBJ	4880*KB*	28.7%	material only
	FBX	1127*KB*	83.5%	material only

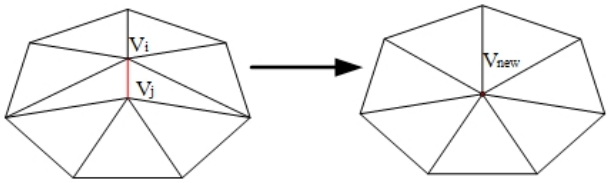


Fig. 4. Schematic diagram of edge collapse

4. OnInstanceEnd () marks the completion of instance loading. In OnInstanceEnd (), you need to write a program fragment that links to the MySQL database to obtain the attribute information of the model. In addition, it is necessary to obtain the uuid of the element to ensure the unique association with the attribute table.

5. Finish () is used at the end of the export to complete the operation of writing the *.gltf file and writing the bin file.

3.2. Geometric lightweight based on the QEM

After removing models redundant data by using digital-analog separation. In order to further lightweight processing of the models. This paper also considers using mathematical method to reduce the amount of triangular surface data of the models. Among mesh model simplification algorithms, geometric delete method is the most widely used method, including vertex deletion, edge collapse and triangle collapse. The QEM algorithm is one of the edge collapse methods, which is fast in operation, occupies less memory, and has better simplification quality.

3.2.1. Edge Collapse algorithm

The basic principle of edge collapse algorithm is shown in Fig. 4. Assuming the edge (V_i, V_j) has the smallest error measure. And Two vertices of the edge are merged to form a new vertex V_{new} to delete the edge (V_i, V_j). Then, the points previously connected to V_i and V_j are connected to the V_{new} point to form a new mesh topology. In this way, one edge collapse is completed, which reduces 2 triangular mesh faces compared to the previous one.

3.2.2. QEM algorithm

A key problem for edge collapse is how to define an error measure to minimize the error between the collapsed model and the original model [13]. In response to this problem, a large number of scholars have studied it and derived a variety of methods. Among them, the Quadratic Error Measure (QEM) algorithm proposed by Garland and Heckbert [14] stands out. The algorithm has the advantages of simple calculation, fast simplification and excellent simplification effect. Its basic idea is to take the square sum of distance from the new vertex to the original adjacent surface as the error measure. The specific algorithm is as follows:

1. Setting parameters. The edge to be folded is set to $\left(V_i, V_j\right)$. The newly generated vertex is $V_{\text{new}} = \begin{bmatrix} x & y & z & 1 \end{bmatrix}^T$. The set of all triangular faces associated with a point is $P(V)$, where represents each triangular face, and it is expressed as $\begin{cases} ax+by+cz+d=0 \\ a^2+b^2+c^2=1 \end{cases}$. Q is the initial error matrix of all vertices. $\Delta(V_{\text{new}})$ is the error measure.

2. Calculating the error measure.

$$\begin{aligned} &\Delta(V_{\text{new}}) = \Delta[\text{x y z 1}]^T \\ &= \sum_{P\in P(V_i)} \left(P^T V_{\text{new}}\right)^2 + \sum_{P\in P(V_j)} \left(P^T V_{\text{new}}\right)^2 \\ &= V_{\text{new}}^T \left\{ \sum_{P\in P(V_i)} \left(PP^T\right) + \sum_{P\in P(V_j)} \left(PP^T\right) \right\} V_{\text{new}} \\ &= V_{\text{new}}^T \left\{ \sum_{P\in P(V_i+V_j)} K_P \right\} V_{\text{new}} \end{aligned} \tag{1}$$

Where $K_P = PP^T = \begin{pmatrix} a^2 & ba & ca & da \\ ab & b^2 & cb & db \\ ac & bc & c^2 & dc \\ ad & bd & cd & d^2 \end{pmatrix}$ is the quadratic fundamental error matrix of the triangular surface P. So $Q = \sum_{P\in P(V)} K_p$ is the quadratic error matrix of point V .

3. Analyzing the factors that affect the error measure values. It is worth noting that the error measure values of the initial vertices of the model are 0 [15], so the cost of edge collapse is $\Delta(V_{\text{new}}) = V_{\text{new}}^T \left(Q_i + Q_j\right) V_{\text{new}}$. It can be seen that the error measure value is determined by the position of the new vertex and the error matrix value of the two vertices of the initial edge.

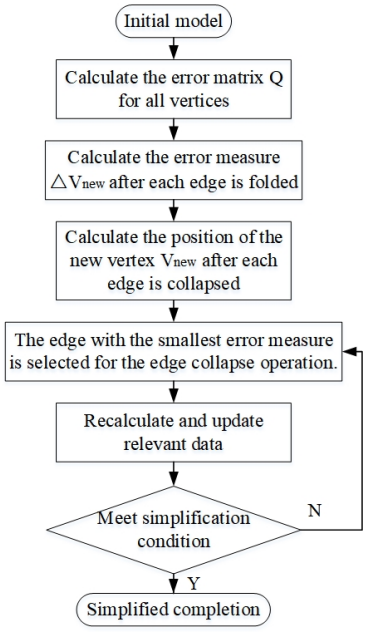


Fig. 5. The flow chart of the QEM algorithm

The flow of the QEM algorithm is shown in Fig. 5, and the iteration stops when the simplified condition is met. The simplification condition is generally to reduce the number of edges to a certain value.

3.3. Lightweight based on Draco

Draco is an open source 3D graphics compression library developed by Google, which aims to improve the storage and network transmission efficiency of 3D models. Because it is dedicated to 3D geometric grids and point cloud data, it has better compression performance than some traditional compression techniques such as ZIP compression file. Draco not only supports compression points, texture, color and all information related to geometry, but also supports OBJ, glTF, drc and other 3D graphics file formats. It compresses the file size greatly on the premise that the models information is not lost. So the models can be loaded to the browser faster, and users can download applications faster [16].

4. Development and implementation of visualization application based on web browsers

WebGL technology enables cross-platform rendering and display of 3D models, while allowing users to interact on the browsers without installing plug-ins [17]. Although WebGL technology has enabled the transition from standalone application development in C++ (or C language) to providing a JavaScript API, the WebGL API contains a lot of knowledge of the most basic computer graphics [18].

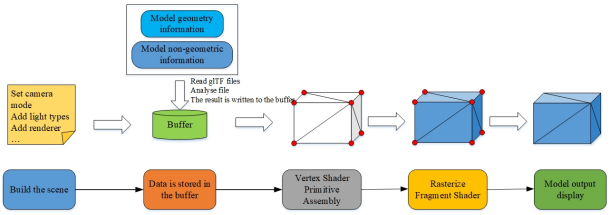


Fig. 6. The visualization process of glTF file based on WebGL

This leads to excessive requirements on the computer level of the developer. Therefore, in order to reduce the difficulty of development and shorten the development cycle, this paper selects the Three.js framework as the rendering engine of this visual application.

The web-based visualization application development of BIM models is mainly divided into two steps: three-dimensional scene framework construction, lightweight file parsing and reading [19]. The visualization process of glTF file based on WebGL is shown in Fig. 6. In this paper, the visualization application of the signal BIM models is based on the Three.js framework. The development of the Three.js framework and the realization of the dynamic interaction of the models are completed in the Visual Studio Code environment.

4.1. Development of Three.js engine

The 3D rendering engine of signal BIM models is developed based on HTML5 technology and Three.js framework. The development process is shown in Fig. 7.

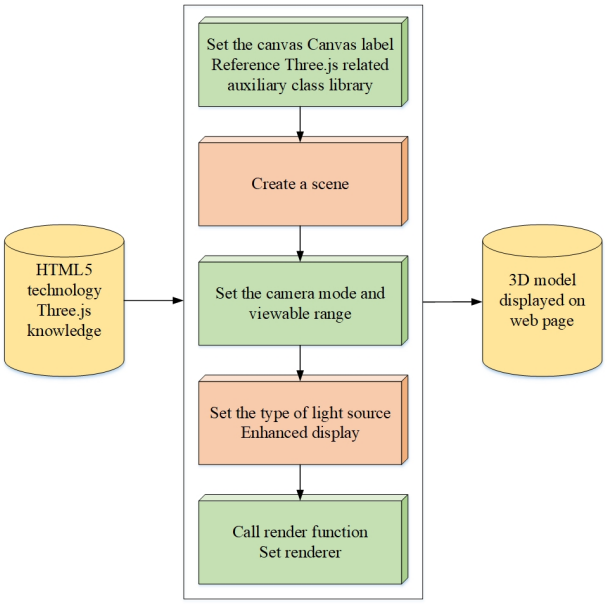


Fig. 7. The development process of Three.js

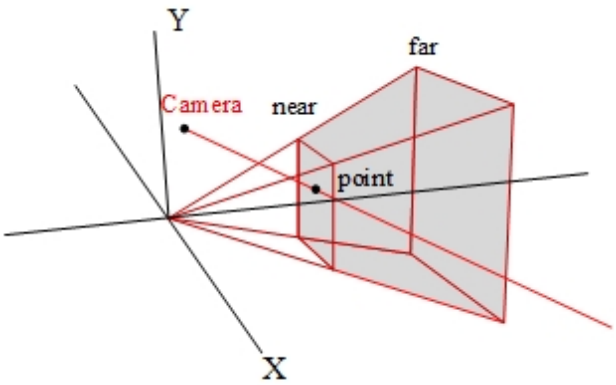


Fig. 8. Schematic diagram of Raycaster ray method

4.2. Model dynamic interaction

Firstly, in order to make the models move and allow users to browse the models in all aspects, the orbitControls method provided by Three.js is used to realize basic functions such as zooming, rotating the models.

Secondly, BIM models often carries a large amount of information. In order to realize that the user can pick up the models and query attribute information, this paper adopts the method of combining the mouse click event with the Raycaster ray method. At the same time, the picked model is also highlighted to increase the experience effect. As shown in Fig. 8, the Raycaster ray method takes the camera as the starting point and emits a ray perpendicular to the screen. Then the result of the collision between the ray and the model is taken as the return value. If there is a collision, the model component that collides with the ray first is returned. If there is no collision, the return value is empty.

5. Experimental verification and performance analysis

This paper takes the models of starting signal, distribution cabinet and signal equipment room with different complexities as examples to verify the feasibility of the proposed lightweight visualization application scheme of signal BIM models based on glTF format and WebGL technology.

1. First of all, this paper creates the signal BIM models through Revit 2018, as shown in Fig. 9.

2. Then the models are lightweighted by Revit secondary development API. The model geometry and semantic information are stored in the glTF file and MySQL database respectively. For the lightweight of internal data, the proposed method is compared with the method directly converted to glTF format, and the results are shown in Table 2.

It can be seen from Table 2 that whether the digital-analog separation is adopted, the data volume of the models is effectively reduced. However, through comparison, it is found that the lightweight effect of the digital-analog separation processing mode is better. In addition, when the complexity of the model is high, the lightweight effect of direct conversion is greatly reduced. Therefore, the lightweight effect of the digital-analog separation processing models is more stable. At the same time, the attribute information is stored in the database, which is also conducive to the operation of adding, deleting, checking, and modifying the information according to the actual situation.

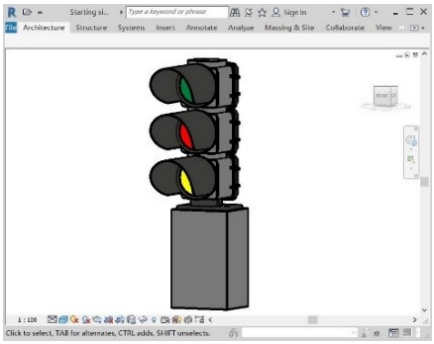
For the lightweight of external data, the starting signal is taken as an example for detailed description. The simplified results are shown in Fig. 10 and in Table 3. In addition, taking the distribution cabinet as an example, this paper compares the application of the QEM algorithm in different file formats, and the results are shown in Table 4. The external lightweight of the signal equipment room is similar, so I won't go into details.

It can be seen from Fig. 10 and Table 3 that with the reduction of the number of triangular faces, the size of the model file has also been reduced to a certain extent. It achieves a better simplification effect while ensuring the visual characteristics of the model. However, when the degree of simplification is 50%, the model has obvious distortion. Through continuous experiments, it is found that when the degree of simplification is between 20% and 35%, the best effect is achieved. In this paper, the degree of simplification is selected as 20%. It can be seen from Table 4 that the application effect of the QEM algorithm on the three file formats is not much different. However, by comparing Tables 1 and 4 (section 3.1), it can be found that the file size of glTF becomes the smallest, which further confirms the good effect of digital-analog separation.

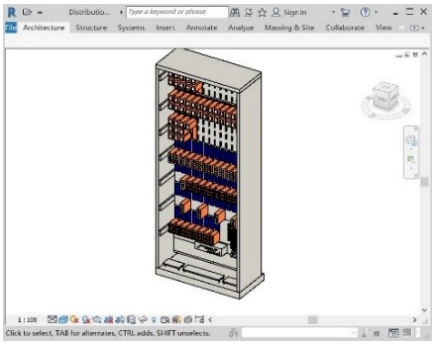
The results of file compression lightweight are shown in Table 5. All lightweight results are shown in Table 6. It can be seen from Table 5 that the Draco algorithm has an excellent compression and lightweight effect. In addition, as the model complexity increases, the lightweight effect also increases. For the most complex model of signal equipment room, the compression rate reaches 90.7%, which effectively improves the transmission capacity of the model. It can be seen

Table 2. Comparison of internal data lightweight

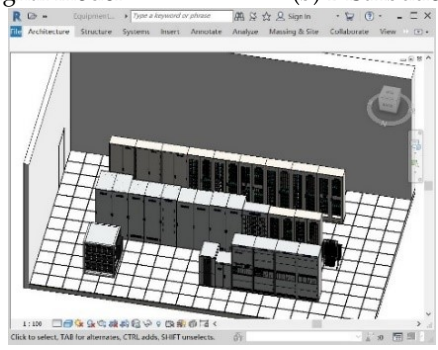
Model	Initial size	Direct conversion (Lightweight rate)	Digital-analog separation (Lightweight rate)
Starting signal	2600 KB	711 KB(72.7%)	382 KB(85.3%)
Distribution cabinet	6840 KB	1687 KB(75.3%)	782 KB(88.6%)
Signal equipment room	260,644 KB	171,756 KB(34.1%)	65,905 KB(67.5%)



(a) Starting signal model

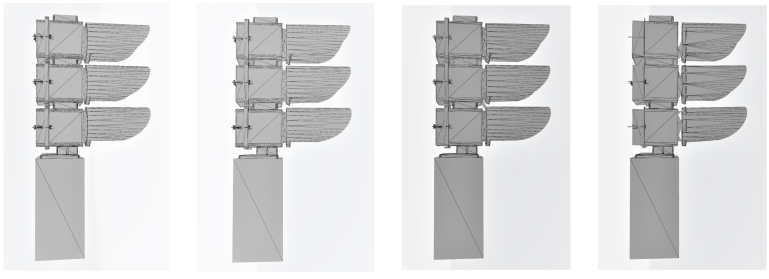


(b) Distribution cabinet model



(c) Signal equipment room model

Fig. 9. Examples of signal BIM models



(a) Original model (b) Simplify 20% (c) Simplify 35% (d) Simplify 50%

Fig. 10. Comparison of the simplified results of the starting signal model

Table 3. Comparison of model simplification data

Mode1	Number of triangles	Simplified	File size
Original	18281	–	372 KB
Simplify 1	14440	20%	324 KB
Simplify 2	11267	35%	283 KB
Simplify 3	7772	50%	239 KB

Table 4. Lightweight comparison of distribution cabinet models

Simplified	Number of triangles	Convert format/ file size		
		g1TE	OBJ	FBX
–	42223	782 KB	4880 KB	1127 KB
20%	33733	681 KB	4299 KB	992 KB
35%	26445	602 KB	3761 KB	868 KB
50%	18372	510 KB	3135 KB	735 KB

from Table 6 that the overall lightweight rate of the model is as high as 98 % and above, indicating that the lightweight scheme proposed in this paper has excellent effect.

3. On this basis, the lightweight processed models are connected to the developed Three.js engine for web-based visual application display. The model interaction operation is shown in Fig. 11, and the model attribute information query is shown in Fig. 12.

4. Finally, the performance of the visualization platform built in this paper is tested from three aspects: frames per second, compatibility and computers with different configurations. Frames Per Second (FPS) refers to the number of frames per second transmitted by an image, which is an intuitive manifestation of WebGL calling local GPU resources to render images. The higher the frame rate value, the smoother the image loading will be. Generally, when the frame rate is lower than 24FPS, the human eye will feel more obvious stuttering. The computer configuration developed this time is as follows: the processor is 11th Gen Intel(R) Core(TM) i5-11400H; the memory is 16.0GB; the graphics card is NVIDIA GeForce RTX 3050. The frame rate test results are shown in Table 7. It can be seen from Table 7 that under this computer configuration, the frame rate is higher than 24PFS and stable at around 60FPS, with excellent rendering efficiency and visual experience.

   The browser is the carrier of the web display, and the compatibility with different browsers affects the user's experience. This paper takes the signal equipment room model as an example to conduct compatibility tests on three mainstream browsers: Google, Firefox and Sogou. The test results are shown in Table 8 . It can be seen from Table 8 that the proposed method has good compatibility on different browsers. But in contrast, Google's compatibility is the best.

   Take the distribution cabinet model as an example, and test it on computers with different hardware configurations. The hardware configuration of the computer is shown in Table 9. The lightweight visualization application effect is shown in Fig. 13. The interactive function and frame rate results are shown in Table 10.

   It can be seen from Fig. 13 and Table 10 that the proposed method has excellent application effect on lightweight visualization on different computers.

6. Conclusion

1. This paper selects the glTF format, which is very suitable for loading and rendering on the Web side, as the target data format. And by designing the content of the Revit secondary development interface, the direct conversion of the RVT model source file to the glTF format is realized, so that the signal BIM model is no longer limited by professional software.

2. At the same time of file format conversion, aiming at the problem of a wide variety of models and large amount of data volume, this paper studies the internal data, the appearance geometry and the file compression. Without affecting the use effect, the data volume of the model is greatly reduced.

3. Combined with HTML5 and WebGL technology, the Web-3D model display functional module was developed to realize the lightweight display of the signal BIM model and the interactive operation of users.

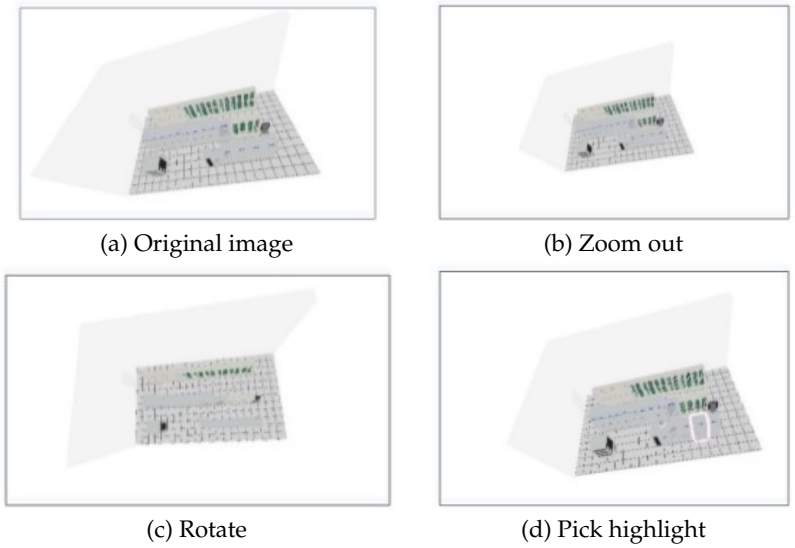
4. Through example verification, this paper has successfully completed the lightweight visualization application of signal BIM model based on glTF format and WebGL technology. However, this paper does not consider the rendering optimization of lightweight models on the browser side. The next step will consider using algorithms such as LOD and visual culling to optimize rendering, in order to reduce the load on the GPU and further improve the loading rate and rendering efficiency. In addition, the application of signal BIM lightweight in the whole life cycle of railway signal engineering will be considered to promote the application depth of BIM technology in railway signal informatization construction.

Table 5. Comparison of compressed results

Model	Before compression	After compression	Compression ratio
Starting signal	324 KB	42 KB	87.0%
Distribution cabinet	681 KB	76 KB	88.8%
Signal equipment room	57351 KB	533 KB	90.7%

Table 6. Overall lightweight comparison of the three models

Model	Initial size	Digital-analog separation	QEM algorithm	After compression	Lightweight rate
Starting signal	2600 KB	382 KB	324 KB	42 KB	98.4%
Distribution cabinet	6840 KB	782 KB	681 KB	76 KB	98.9%
Signal equipment room	260, 644 KB	65, 905 KB	57351 KB	533 KB	99.7%



(a) Original image (b) Zoom out

(c) Rotate (d) Pick highlight

Fig. 11. Schematic diagram of model interaction

Table 7. Comparison of model rendering frame rate

Model type	Frames Per Second
Starting signal	61 FPS $(60-144)$
Distribution cabinet	59 FPS $(59-144)$
signal equipment room	60 FPS $(56-97)$

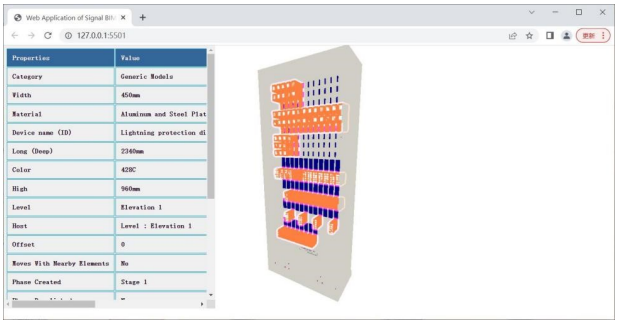


Fig. 12. Display of model attribute information

Table 8. Results of browser compatibility test

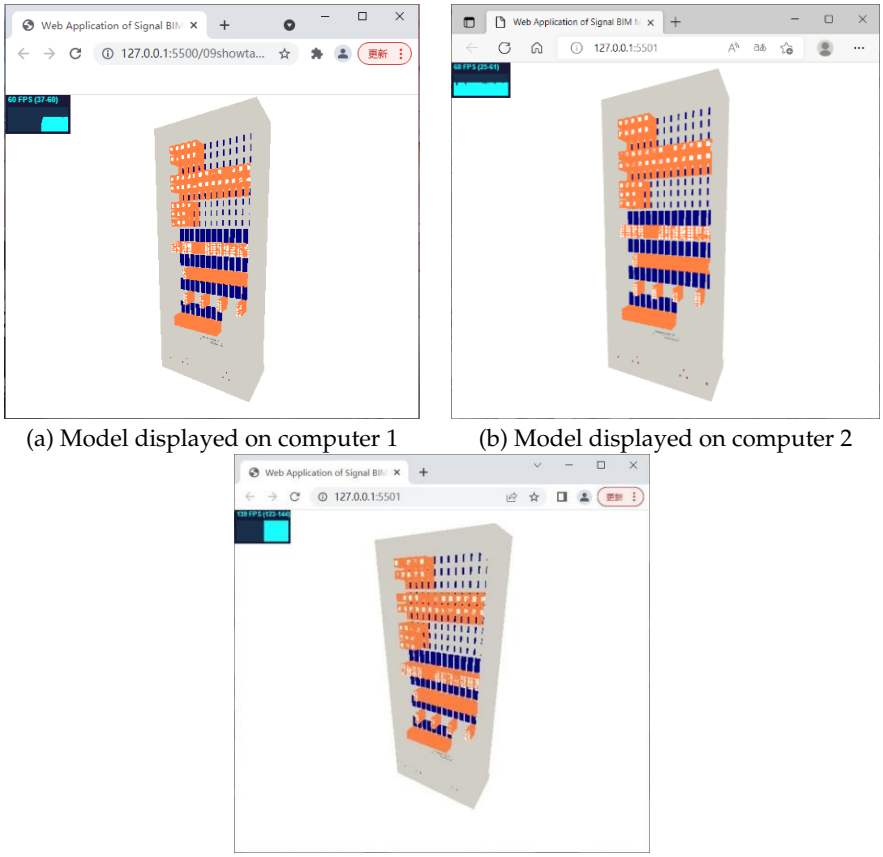
Browser	Interactive effect	FPS	Overview
Chrome	Well	73FPS (60-96)	1
Firefox	Well	45FPS (41-63)	3
Sogou	Well	66FPS (46-119)	2

Acknowledgments

Granted by Gansu Science and Technology Plan (Key R & D Program) (Grant No.: 20YF8GA037).

Table 9. Configuration of different computers

Computer type	Processor	Memory	Graphics card
Computer 1	Intel(R) Core (TM) i5-5200U	4.0 GB	AMD Radeon (TM) R5 M320
Computer 2	Intel(R) Core (TM) i5-7200U	8.0 GB	NVIDIA GeForce 940MX
Computer 3	11th Gen Intel(R) Core (TM) i5-11400H	16.0 GB	NVIDIA GeForce RTX 3050



(a) Model displayed on computer 1 (b) Model displayed on computer 2

(c) Model displayed on computer 3

Fig. 13. Models displayed on computers with different configurations

Table 10. Test results for different computer configurations

Computer	Interactive effect	Frames Per Second
Computer 1	Well	60FPS (37-60)
Computer 2	Well	60FPS (25-61)
Computer 3	Well	139FPS (123-144)

References

[1] M. Li, X. Xiong, and Q. Yin, (2021) *"Smart City Construction Visual Simulation Application Based on Intelligent BIM Technology"* **International Journal of Pattern Recognition and Artificial Intelligence** *35*(13): 2155014. DOI: 10.1142/S0218001421550144.

[2] W. Liu, (2019) *"The practice and thinking of BIM technology in digital railway construction"* **Journal of the China Railway Society** (3): 97–101.

[3] P. Zhi. *"Research on BIM-based railway construction management platform and key technologies"*. (phdthesis). Beijing: China Academy of Railway Sciences, 2018.

[4] C. Ren, Y. Zang, and Z. Liu, (2021) *"Visualization Research on Railway Signal Operation and Maintenance Platform Based on Cesium Engine"* **Railway Standard Design** (07): 172–178.

[5] J. Liu, Q. Su, J. Chen, Y. Pei, and M. Dong, (2022) *"Research on visualization of railway operation and maintenance based on WebGL extension model"* **Journal of Railway Science and Engineering** (04): 892–900.

[6] C. Jin and Z. Li, (2021) *"Research on the design and key technologies of BIM construction and maintenance platform for the four railways"* **Journal of Railway Engineering Society *38***(09): 93–99.

[7] N. Wang, S. Wang, and W. Zeng, (2021) *"WebGL-based Rail Transit BIM Lightweight Application Scheme"* **Railway Computer Application *30***(10): 30–34.

[8] W. Luo and C. Qiu. *Revit 2018 architectural design: from introduction to mastery (Chinese version)*. Beijing: China Machine Press, 2018, 205–207.

[9] W. Zhang. *"Research on the lightweight visual operation and maintenance management system of urban integrated pipe gallery based on BIM"*. (mathesis). Xi'an University of Architecture and Technology, 2020.

[10] J. Lv, H. Jin, J. Tan, and P. Wang, (2020) *"The application of glTF in the lightweight of BIM model"* **Technology Innovation and Application *6***: 174–176.

[11] Z. Xu, L. Zhang, H. Li, Y.-H. Lin, and S. Yin, (2020) *"Combining IFC and 3D tiles to create 3D visualization for building information modeling"* **Automation in Construction *109***: 102995. DOI: 10.1016/j.autcon.2019.102995.

[12] X. Hu, J. Chen, D. Yang, and Y. Zhu, (2021) *"Research on BIM+WebGIS Fusion Application Based on Revit Secondary Development"* **Journal of Central South University (Science and Technology) *52***(11): 3930–3942.

[13] F. Dong, R. Zhang, Y. Liu, and X. Zhou, (2007) *"A Survey of Simplification Algorithms for Mesh Models"* **Journal of China Three Gorges University (Natural Sciences) *29***(1): 54–59.

[14] M. Garland and P. S. Heckbert. "Surface simplification using quadric error metrics". In: *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. 1997, 209–216.

[15] S. Sun, M. Zhang, and Z. Gou, (2015) *"Smoothing algorithm for planar and surface mesh based on element geometric deformation"* **Mathematical Problems in Engineering** (1): 1–9. DOI: 10.1155/2015/435648.

[16] *Draco Compressed Meshes with glTF and 3D Tiles*. 2018. URL: https://cesium.com/blog/2018/04/09/draco-compression/.

[17] D. Liu, J. Peng, Y. Wang, M. Huang, Q. He, Y. Yan, B. Ma, C. Yue, and Y. Xie, (2019) *"Implementation of interactive three-dimensional visualization of air pollutants using WebGL"* **Environmental modelling & software *114***: 188–194. DOI: 10.1016/j.envsoft.2019.01.019.

[18] Z. Liu, X. Gu, Q. Dong, S. Tu, and S. Li, (2021) *"3D visualization of airport pavement quality based on BIM and WebGL integration"* **J. Transp. Eng. Part B Pavements *147***: 04021024. DOI: 10.1061/JPEODX.0000280.

[19] Z. Xu, X. Xu, Q. Li, and X. Zhang, (2016) *"Visualization Analysis of Building Information Model Based on WebGL and IFC"* **Journal of Central South University (Science and Technology *46***(2): 444–449.