Research and Implementation of Real-time Render Optimization Algorithm Based on GPU

Songlan Wang*, Ji Zhang
Nanjing University of Chinese Medicine, Nanjing, China
*Corresponding author’s e-mail: nelsonwang2001@icloud.com

Abstract. In the continuous optimization of computer graphics, game engine and virtual reality have become the focus of research and innovation in the current technology field. Rendering is a core technology to show a variety of graphic effects, which has been attached great importance to and applied by the whole society. Nowadays, although the rendering of large-scale and complex scenes has a wide range of applications, the actual optimization requirements are very high. Therefore, in the future technology application and research and development, based on higher and higher technical requirements, it is necessary to improve the efficiency of actual rendering while ensuring or improving the rendering quality. Therefore, this paper studies how to use the algorithm to improve the efficiency of actual rendering, so as to provide a new basis for future computer graphics research.

1. Large-scale rendering of complex scenes
Under normal circumstances, can choose when rendering large-scale complex scenarios, there are many ways of eliminating algorithm such as sunscreen, and instantiate technology, LOD algorithm, and this paper studies on the basis of this puts forward a new rendering algorithm, which is under the guidance of the Unity engine, terrain and its model in optimization design of complex scenes. Combined with the current technology application and research and development situation, this new algorithm design has a strong efficiency and correctness, even in large and complex scenes can still improve the efficiency and quality of the actual rendering[1-4]. In simple terms, rendering refers to the process of using software to generate images from a model during computer drawing. In other words, the GPU determines the color change of each pixel on the screen after the model data is fed into the rendering pipeline. In this process, geometry, texture, viewpoint, illumination and other contents need to be used to represent the state of three-dimensional objects in the model. As shown in Figure 1 below, this is the real-time rendering pixel filling process in this study:

![Fig. 1 Pixel filling process](image-url)
Based on the analysis of the above figure, it can be seen that the coordinate system of the 3D object is used to project it onto the 2D screen, and the coordinate space of the model constructed belongs to the object space. During rendering, the transformation of the coordinate system will be used to show the scaling and rotation of the graph, as shown in Figure 2 below:

**Fig. 2** Transformation operations and corresponding spaces

Both the world transformation and the view transformation need to use the most basic scaling, rotation and translation transformation to complete the relevant calculation, in which the specific three-dimensional scaling transformation can be represented by the following $4 \times 4$ matrix, where $S_x, S_y$ and $S_z$ refer to the scaling coefficients in different directions[5-8].

\[
\begin{pmatrix}
S_x & 0 & 0 & 0 \\
0 & S_y & 0 & 0 \\
0 & 0 & S_z & 0 \\
0 & 0 & 0 & 1
\end{pmatrix}
\]  \hspace{1cm} (1)

Combined with the analysis of 2D rotation operation process, a new vector $P'$ is formed after the Angle of rotation $\theta$ of vector $P$. Then $P'(x', y')$ can be obtained by combining the analysis of coordinate system formula as follows:

\[
x' = r \cos(\phi + \theta) = r \cos \phi \cos \theta - r \sin \phi \sin \theta = x \cos \theta - y \sin \theta
\]

\[
y' = r \sin(\phi + \theta) = r \cos \phi \sin \theta - r \sin \phi \cos \theta = x \sin \theta - y \cos \theta
\]  \hspace{1cm} (2)

It can also be combined with matrix multiplication to express vector form, as follows:

\[
\begin{pmatrix}
x' \\
y'
\end{pmatrix} =
\begin{pmatrix}
\cos \theta - \sin \theta \\
\sin \theta \cos \theta
\end{pmatrix}
\begin{pmatrix}
x \\
y
\end{pmatrix}
\]  \hspace{1cm} (3)

Thus, the 3D rotation operation needs to be defined correctly on the basis of a clear rotation axis. Wherein, the result of $(x, y, z)$ rotation is $(x', y', z')$, and $z=z'$, so the vector form of the above formula is expressed by matrix multiplication:

\[
\begin{pmatrix}
x' \\
y' \\
z'
\end{pmatrix} =
\begin{pmatrix}
\cos \theta - \sin \theta & 0 & 0 \\
\sin \theta \cos \theta & 0 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
x \\
y \\
z
\end{pmatrix}
\]  \hspace{1cm} (4)

In the translation operation, $(x, y, z)$ is transferred to $(x+1, y+1, z+1)$, combined with the vector addition formula to calculate, the following results can be obtained:

\[
\begin{pmatrix}
x \\
y \\
z
\end{pmatrix} +
\begin{pmatrix}
t_x \\
t_y \\
t_z
\end{pmatrix} =
\begin{pmatrix}
100t_x \\
010t_y \\
000t_z
\end{pmatrix}
\]  \hspace{1cm} (5)
At this time, when the world space is transferred to the camera space, the space transformation can be carried out according to the coordinate system of the two so as to meet the operational requirements of visual transformation. The coordinate systems at this time are \([O, E1, E2, E3]\) and \([\text{EYE}, U, V, N]\) respectively. When moving the camera space, let the EYE move to the position of O. At this time, the vector can be expressed as (-eye, -eye, -eye), and the specific matrix RT is shown as follows:

\[
T = \begin{pmatrix}
100 - \text{EYE}_x \\
010 - \text{EYE}_y \\
001 - \text{EYE}_z \\
0001
\end{pmatrix}
\]  

(6)

In the case that camera space and world space overlap, translation transformation and rotation transformation can be used to obtain the corresponding camera space, and the specific matrix R is shown as follows:

\[
R = \begin{pmatrix}
u_x & u_y & u_z & 0 \\
v_x & v_y & v_z & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1
\end{pmatrix}
\]  

(7)

Combined with the above two formulas, the visual transformation matrix MVVIEW can be obtained as follows:

\[
M_{\text{view}} = R \cdot T
\]

\[
= \begin{pmatrix}
u_x & u_y & u_z & 0 \\
v_x & v_y & v_z & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1
\end{pmatrix} \begin{pmatrix}
100 - \text{EYE}_x \\
010 - \text{EYE}_y \\
001 - \text{EYE}_z \\
0001
\end{pmatrix}
\]

\[
= \begin{pmatrix}
u_x & u_y & u_z & -\text{EYE}_x \cdot u \\
v_x & v_y & v_z & -\text{EYE}_y \cdot v \\
0 & 0 & 0 & -\text{EYE}_z \cdot n \\
0 & 0 & 0 & 1
\end{pmatrix}
\]  

(8)

2. Optimization algorithm

2.1. Elimination algorithm

This algorithm involves three aspects: the first, the scene body elimination. As a three-dimensional space, vertices will be transferred from the observation space to the clipping space in order to better cut and render the primitives. The primitives completely inside the viewing body will be effectively saved, while the other primitives will be completely eliminated. Similarly, the primitives intersecting with the boundary of the viewing body will also be clipped out. This work is mainly used in the first processing phase of the rendering pipeline, followed by the input geometry phase and rasterization phase, which refers to the application phase, as well as the output after processing the vertex data and render data. Generally speaking, on the basis of clear view position, no matter how large the complex scene is, it only has a small view range, and the
GPU value needs to transfer this part to the rendering pipeline to draw the screen, without the need for complete scene rendering. It should be noted that during the roaming period, as the viewpoint position keeps changing, the rendered scene data will also change accordingly, which can effectively avoid excessive computing tasks and improve the efficiency of actual rendering. Second, occlusion culling. This method is the most common forms of address design Z-buffer, but the efficiency of this algorithm in practical application is not strong, only strict visible objects of the scene inside the real-time rendering, and is located in the last stage of the rendering pipeline, therefore the presence too much not visible objects in the scene, then can appear a lot of unnecessary computing tasks. In order to solve this problem, the staff need to pursue efficient calculation at the same time, but also priority to choose more accurate elimination, in order to prevent the final picture of the existence of redundant calculation process; Third, the back is removed[9-10]. In the rendering of more complex scenes, after the fixed view range is defined, the opaque objects contained in the scene need to be observed from this area, while the polygon data on the back of the object need to be removed without relevant processing, so as to improve the actual rendering efficiency on the basis of reducing the amount of calculation. Figure 3 shows the actual application elimination process:

![Elimination flow chart](image)

**Fig. 3** Elimination flow chart

2.2. LOD algorithm

Similar to the above eliminating algorithm, this kind of algorithm first appeared in the mid-1970s and was proposed by J. Clarke. In practical operation, different precision grids should be used to render and optimize the rendering of objects with different view distances. However, due to the unclear working logic of early computers, it did not receive much attention. With the current technological innovation and exploration, this theory has brought a new way of thinking for subsequent technological development. At present, this kind of algorithm is mainly divided into three kinds: first, viewpoint dependent LOD algorithm; Second, discrete LOD algorithm; Third, continuous LOD algorithm. Figure 4 is the actual operation flow chart:
2.3. Geometry clipmap algorithm

In a massive wave of complex terrain rendering work, need to use a lot of rendering resources and memory space to deal with the related data, and choose the above two algorithms may demand that accords with a condition, but need at run time to build and modify the vertex and index buffer, such not only can aggravate the pressure calculation, and affect the efficiency of the GPU. With such an algorithm, a set of nested regular grids can be used to store the relevant terrain data and ensure that it can be changed with the continuous innovation of the viewpoint position. The regular mesh used in this case is simpler and more effective than the irregular mesh and has a stable rendering rate in practice. For example, terrain data is stored as a 2D height map in the framework designed by the algorithm, which can be filtered into a mipmap pyramid containing L-layer after effective processing[11-12].

3. The unity engine

In the background of the rapid development of computer technology and graphics, system users have higher and higher requirements for the quality of 3D virtual animation. At this time, most of the network giants began to strengthen the research and development of relevant technologies, and thus put forward a new game engine. Unity engine as a kind of comprehensive game development engine, can complete a variety of interactive platform construction, such as architectural visualization, real-time 3 d simulation animation and 3 d games and so on, and effective integration in the practical application of rendering, audio, and graphics and other functions, can be compatible with most of the current market mainstream 3 d software and art resource file format. In practical operation, the staff does not need to understand the complex algorithm logic contained in it, but only needs to use the simple and safe Boo or C# and other high-level languages to get the original complex program functions.

4. Case design analysis

At present, in the application process of complex scene terrain rendering, the most common technical means is to use the continuous level details of quadtree and real-time optimization adaptive mesh to complete the data consolidation and rendering of outdoor ground type. The specific design steps are divided into the following points: first, on the basis of clear application of quadtree and octree and other contents, starting with the scene anatomy and analysis; Secondly, the location of the camera is used to sort out and check the nodes of the tree structure, and the hierarchy structure of the surrounding body is used to directly and effectively remove the invisible polygons. Thirdly, the LOD algorithm technology studied and analyzed in this paper is used to update the relevant information of the existing viewpoint and the distance of the object in real time, and automatically select the appropriate technical requirements for the scene object.
Fourthly, according to the habit of human eyes to observe the world, low-precision mesh is applied to represent distant objects, while high-precision mesh is used to represent near objects. In the end, the vertex value of GPU output is effectively reduced and the number of actual rendering is controlled.

In this process, the designer should choose the improved Geometry Clipmap algorithm for research according to the computing advantages and processing power shown by the GPU. Combination of experimental design and analysis of research in this paper, rendering optimization case, large-scale forest scene as the basis, the validation process is divided into horizontal and vertical, need to choose the former before and after the improvement algorithm for the random point location same scene rendering, which need to increase the complexity of the scene to test the algorithm robustness.

In the validation of the effectiveness and accuracy of the algorithm, the horizontal comparison analysis is the application performance differences of different algorithms. Combined with the actual changes analysis, it can be seen that compared with the traditional technology form, the improved algorithm can more effectively process and render complex scenes, and can improve the practical rendering performance. In the longitudinal comparison, different complexity and data size should be used to verify the algorithm, so as to clarify the robustness of the selected algorithm. The final results show that with the increase of the complexity of the scene and the increase of the actual processing data, the application performance of the actual algorithm is higher. Therefore, it is proved that the improved Geometry Clipmap algorithm has more application value and can deal with scenarios of different complexity and data size.

5. Conclusion

To sum up, under the background of the new era, with the continuous expansion of the application range of 3D virtual images, the requirements of the virtual reality system constructed in various fields are increasingly high. At this time, in the face of large-scale complex scene rendering requirements, based on the current computer graphics processing skills, research on how to use algorithms to improve the actual rendering efficiency and quality, is the focus of current technology research and development. Therefore, in the future technology research and development innovation, researchers should comprehensively improve the real-time and effectiveness of rendering operation on the basis of guaranteeing rendering effect, which will play a positive role in the development of computer graphics. At the same time, it is necessary to strengthen the training of professionals, and pass on to them the technical knowledge and professional quality with the characteristics of The Times and advanced nature, so as to encourage them to actively participate in related research and development activities, and find more valuable research content from them.

References

[1] Dubuisson, F., Rezkallah, M., Ibrahim, H., et al. (2021) Real-Time Implementation of the Predictive-Based Control with Bacterial Foraging Optimization Technique for Power Management in Standalone Microgrid Application[J]. Energies, 14.

[2] Gong, X., Le, Z. (2020) Research and implementation of multi-object tracking based on vision DSP[J]. Journal of Real-Time Image Processing, (3).

[3] Li, S., Li, W., Zhang, H., et al. (2019) Research and implementation of parallel artificial bee colony algorithm based on ternary optical computer[J]. Automatika, 60(4):423-432.
[4] Wang, M., Zhang, H., Hu, Q., et al. (2019) Research and implementation of a non-supporting 3D printing method based on 5-axis dynamic slice algorithm[J]. Robotics and Computer-Integrated Manufacturing, 57(JUN.):496-505.

[5] Larijani, M. R., Zolghadri, M. R. (2021) Design and implementation of an ADC-based real-time simulator along with an optimal selection of the switch model parameters[J]. Electrical Engineering, (2).

[6] Forster, D., Inderka, R. B., Gauterin, F. (2019) Data-Driven Identification of Characteristic Real-Driving Cycles Based on k-Means Clustering and Mixed-Integer Optimization[J]. IEEE Transactions on Vehicular Technology, PP(99):1-1.

[7] Zhang, G., Kou, X. (2021) Research and implementation of digital 3D panoramic visual communication technology based on virtual reality[J]. International Journal of Communication Systems.

[8] Yan, Y., Yuan, L., Li, Y. (2020) Research on Optimization of Production Decision Based on Payment Time and Price Coordination[J]. Complexity.

[9] Virgili-Llop, J., Zagaris, C., Zappulla, R., et al. (2019) A convex-programming-based guidance algorithm to capture a tumbling object on orbit using a spacecraft equipped with a robotic manipulator[J]. The International journal of robotics research, 38(1):40-72.

[10] Pedersen, C. K., Mackey, J. E, Teytelboym, O. M. (2020) Protocol Optimization and Implementation of Dual-Energy and Dual-Source Computed Tomography in Clinical Practice: Field of View, Speed, or Material Separation?[J]. Journal of Computer Assisted Tomography, 2020, Publish Ahead of Print(4).

[11] Lin, Z., Sinha, S., Zhang, W. (2020) Hard-ODT: Hardware-Friendly Online Decision Tree Learning Algorithm and System[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, PP(99):1-1.

[12] Vázquez-Mourelle, R., Carracedo-Martinez, E., Figueiras, A. (2020) Impact of a health alert and its implementation on flutamide prescriptions for women: an interrupted time series analysis[J]. BMC Health Services Research, 20(1).