Wang et al. EURASIP Journal on Wireless Communications and Networking (2019) 2019:79
https://doi.org/10.1186/s13638-019-1401-9

EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

Performance bottleneck analysis and resource optimized distribution method for IoT cloud rendering computing system in cyber-enabled applications

Ronghe Wang[1], Bo Zhang[1*], Manqing Wu[1], Jun Zhang[3], Xiaolei Guo[1], Xinhai Zhang[1], Huibo Li[1], Dong Jiao[1] and Shilong Ma[2]

Abstract

This paper analyzes current cloud computing, cloud rendering industry, and related businesses. In this field, cloud system performance lacks unified evaluation criterion. A novel analysis method and a related measure of cloud rendering system performance are presented in this paper. The main paper investigates the number of system concurrent users and average response delay about user access, average frame speed, system operation speed, and average response about user browsing system. This paper makes a theoretical analysis of a core business process about cloud rendering system, multi-task rendering processes especially. This paper analyzes the efficiency, average frame rate, rendering performance bottleneck of the cloud rendering system, and put forward a unique parameter adjustment strategy to improve system performance, by optimizing related server and rendering machine configuration. This paper puts forward a method to reduce the bottleneck of the system and prevent system performance deterioration in the new scheme. This paper puts forward a set of system optimization strategies to improve system performance. This is a new cloud rendering system performance optimization configuration scheme and optimization strategy.

Keywords: Mobile network, Big data, Performance bottleneck, Visualization rendering, Performance optimization

1 Introduction

At present, there are a lot of cloud rendering systems in the field of graphics, virtual reality, and computer vision, but these systems lack a unified system performance analysis method [1–5]. 3D rendering has demanding requirements on the hardware configuration and command response. Cloud rendering system faces a large number of rendering requests from users. There are a large number of simultaneous rendering requests, which will increase great pressure to the backend server system. The user sends commands by terminal systems. The servers receive instructions and complete the rendering task immediately, and the rendered image is transmitted to the user terminal at the same time. If the average response time is too slow, it will greatly reduce the user experience.

The performance measure of cloud rendering system needs to examine the following main points: (1) the number of concurrent users, the design target of cloud rendering system is to withstand a certain scale of the user's concurrent access, so the number of concurrent users is an important index. (2) the average response delay, cloud rendering system requires certain time to respond to any operation issued by the user. 3D rendering in general more than 25 FPS the user will feel a smooth screen, so the average response delay is also an important index.

This paper first analyzes the time consumption of task scheduling and rendering distribution. The efficiency and effect of quantization performance were converted into mathematical symbols. Then, the paper analyses the 3D rendering process, especially analysis multi-task

* Correspondence: bozhangcetc@163.com
[1]National Engineering Laboratory for Public Security Risk Perception and Control by Big Data (PSRPC), China Academy of Electronics and Information Technology, Beijing, China
Full list of author information is available at the end of the article

© The Author(s). 2019 **Open Access** This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

rendering process and rendering performance by a mathematical formula. Through rigorous mathematical analysis of the key parameters of the system, the paper calculates the number of concurrent users and the average response delay. In the paper, we will show a detailed study of these key parameters which is how to influence the performance of the system. The paper puts forward a performance parameter adjustment strategy to enhance system performance.

2 Related work

The technology has developed quite mature and was successfully applied to the movie industry, like CG rendering, and 3Ds max scene rendering [6–8]. This type of rendering does not require real-time generally, that is, as long as the rendering begins, and the user only needs to wait for the results to be returned [9–12]. However, it needs to take into account the efficiency of the interaction process for some real-time demands. At present, it lacks a unified evaluation standard and efficiency analysis method of cloud rendering both at home and abroad [13–16]. Although there are few cloud rendering technology in the mobile terminal application, the characteristics of cloud rendering technology have provided a great convenience in mobile migration terminal. Cloud rendering mode is similar to the cloud computing model [17–20], and its main idea is to transfer the user's local 3D rendering work completely to a cloud rendering server which has powerful rendering processing capabilities. The client sends commands to cloud rendering servers [21–26]. The server renderings tasks according to the instructions of users, and the results will be sent back to the user to display [27–31]. The benefits of cloud rendering are that users do not need to worry about the hardware configuration and software compatibility of local equipment [32–35]. All rendering tasks are completed on a cloud rendering server. Although data and development of cloud rendering [36–39] help the user to solve a lot of personal problems, there is a lack of a unified evaluation criteria for its performance and efficiency. As for the parallel task layer of the IoT cloud rendering computing system, each computing node device has an independent parallel task scheduling module. Relying on this module, the node device can no longer focus on the communication details of the rendering application server [40–44]. All scheduling management and operations are encapsulated in a parallel task scheduling module. The purpose of this is to introduce middleware to reduce the coupling between the node device and the rendering system in the server system. The node device can shunt the user's instruction request, improve the operation efficiency of the IoT cloud rendering computing system, and enhance the rendering by interacting with the instruction portability of system and parallel task scheduling modules [45–47].

3 Methods

The definition of time cost (we consider t_0 and t_1 as constants in this section) are as follows: (1) t_0 means a time required that users start an operation and balance loading to resource management server. (2) t_1 means t_0 plus query, get pictures, and return results time. (3) $t_{dispatch}$ means send instructions time. (4) t_{render} means execute the render instruction time at the render machine. (5) t_{upload} means the upload render results to the file server and the database server time.

3.1 Instructions distribution time consumption

The time of the cloud rendering system to distribute commands can be expressed as $t_{dispatch} = t_{dis\_wait} + t_{ask} + t_{send}$. The t_{dis_wait} means the commands waiting time in distribution queue, which can be ignored when the queue is empty. The t_{ask} means time required of scheduling system to query all rendering machine performance status. The t_{send} means the time required of the scheduling system send out commands.

3.2 Rendering command processing time consumption

Rendering instruction processing time can be expressed as $t_{render} = t_{rend\_wait} + t_{scene\_create} + t_{take\_photo}$. The t_{rend_wait} expresses waiting time in render queue, which can be ignored when there is no queuer in the render queue. t_{scene_create} means the time required to execute instructions when a scene is created. t_{take_photo} means the time required to shoot all the pictures.

4 The theory analysis of cloud rendering system business process

In this section, we express the performance of cloud rendering system with corresponding mathematical expressions and find the main contribution of the cloud rendering system. Cloud rendering system business processes mainly include (1) single task non-rendering process. (2) Single task rendering process. (3) Multi-task rendering process.

4.1 Single task rendering process

The single task rendering process is one of the typical processes in the business process field, and the time consumption of each part is as follows:

(1) In $t_{dispatch}$ part, t_{dis_wait} is omitted if there is no wait in queue; t_{ask} is parallel TCP request time. Each part is a long connection, so $t_{ask} = 2(\frac{\Delta_d}{W} + \Delta_w)$; it needs two times to determine whether the instructions communications is a success or not, so $t_{dispatch} = 4(\frac{\Delta_d}{W} + \Delta_w)$.

(2) The t_{render}: t_{dis_wait} is omitted if there is not wait in queue; scene creation process needs consume time $t_{scene\_create} = P\Delta_p$; scene requires time $t_{scene\_create} = P\Delta_p$; finally, $t_{render} = P\Delta_p + C\Delta_c$.

(3) t_{upload} means all pictures uploaded in time. Among them, the TCP long connection time requirement $t_{\text{up\_pic}} = C(\frac{D}{W} + \Delta_w) + \Delta_{tcp}$ and the results upload database time requirement $t_{\text{up\_database}} = \frac{\Delta_d}{W} + \Delta_w + \Delta_{\text{tcp}}$.

So, single task rendering process final time requirement can be presented as $T_{SR} \approx P\Delta_p + C(\Delta_c + \frac{D}{W}) + Q_1$. From this formula, we can see that the effect of T_{SR} are mainly in scene creation time Δ_p, picture shoot time, and transmission time $C(\Delta_c + \frac{D}{W})$.

4.2 Multi-task rendering process

The multi-task rendering process is more complex than single task rendering process. It is mostly consumed time to wait in queue. This paper assumes that there are S tasks to be executed simultaneously, and the total process time T_{MR} can be expressed as

$$T_{MR} = t_0 + t_{\text{dispatch}} + t_{\text{render}} + t_{\text{upload}} + t_1 \tag{1}$$

The time required for all task scheduling, $t_{\text{dispatch}} = \Delta \text{dis}_{\text{wait}\frac{S}{N}} + t_{\text{ask}} + t_{\text{send}} = [\frac{S}{N}](t_{\text{ask}} + t_{\text{send}}) = 4[\frac{S}{N}](\frac{\Delta_d}{W} + \Delta_w)$.

For t_{render} part, because there are multiple render machines parallel rendering, tasks will be evenly distributed to each machine according to task scheduling strategy, and rendering machine mostly receives $\lceil\frac{S}{M}\rceil$ tasks. Because there are M renderer, each renderer task distribution cycle $\overline{t_d}$ is as follows, $\overline{t_d} = \frac{M}{N}(t_{\text{ask}} + t_{\text{send}}) = \frac{4M}{N}(\frac{\Delta_d}{W} + \Delta_w)$. The definition of an average time of task occupied space $\overline{t_o}$ is as follows: $\overline{t_o} = P\Delta_p + C\Delta_c$. In the premise of the above definition, the paper gives an important conclusion. For each renderer, if $K\overline{t_d} \geq \overline{t_o}$ or $S \leq M \cdot K$, the rendering engine will never have tasks waiting in render queue. Therefore, as long as the condition $K\overline{t_d} \geq \overline{t_o}$ or $S \leq M \cdot K$ is established, any tasks that will arrive will be assigned immediately to site for rendering without congestion. On the contrary, if these two conditions are not established, namely, the condition of $K\overline{t_d} < \overline{t_o}$ and $S > M \cdot K$ is established, rendering system congestion will occur.

4.2.1 Multi-task rendering process in blocking status

In the blocking status, the $K\overline{t_d} < \overline{t_o}$ and $S > M \cdot K$ conditions are both established. In the status, the render time t_{r2} is

$$t_{r2}(S,M,K) = \begin{cases} \left(\left\lceil\frac{S}{M}\right\rceil \bmod K-1\right)\overline{t_d} + \left(\left\lceil\frac{S}{M}\right\rceil \text{ div } K + 1\right)\overline{t_o}, & \left\lceil\frac{S}{M}\right\rceil \bmod K \neq 0 \\ (K-1)\overline{t_d} + \left(\left\lceil\frac{S}{M}\right\rceil \text{ div } K\right)\overline{t_o}, & \left\lceil\frac{S}{M}\right\rceil \bmod K = 0 \end{cases} \tag{2}$$

In this paper, $t_s(i)$ represents required time function before each rendering site achieves full load working status. The parameter i indicates the number of rendering site, then $t_s(i)$ is expressed as the following formula:

$$t_s(i) = (i-1)\overline{t_d} \quad (K \geq i \geq 1) \tag{3}$$

Due to the time difference of receiving task in different sites, the time of each site achieve full load working status will be decided by task distribution cycle $\overline{t_d}$. The $t_s(i)$ of each rendering site is not equal and increases with the growth of i, so the number of each site assigned can be defined as $\text{dis}_c(i)$ function. It is expressed as following formula:

$$\text{dis}_c(i) = \left\lceil\frac{S}{M}\right\rceil \text{ div } K + \varepsilon\left(\left\lceil\frac{S}{M}\right\rceil \bmod K - i\right) \tag{4}$$

Among them, $\varepsilon(x)$ is unit step function, which is defined as $\varepsilon(x) = \begin{cases} 0, x < 0 \\ 1, x \geq 0. \end{cases}$

Therefore, the time consumption of each site to complete corresponding rendering tasks, which can be defined as the following:

$$\begin{aligned} t_e(i) &= t_s(i) + \overline{t_o}\text{dis}_c(i) \\ &= (i-1)\overline{t_d} + \left(\left\lceil\frac{S}{M}\right\rceil \text{ div } K + \varepsilon\left(\left\lceil\frac{S}{M}\right\rceil \bmod K - i\right)\right)\overline{t_o} \end{aligned} \tag{5}$$

After all, the sites complete corresponding rendering tasks, the rendering process is over, so the time consumption of completing rendering tasks is shown in the following formula:

$$t_{r2} = \max\left(\cup_{i=1}^{K} t_e(i)\right) \tag{6}$$

In particular, when $\lceil\frac{S}{M}\rceil$ is times of K, this will lead to $\lceil\frac{S}{M}\rceil \bmod K = 0$, because $1 \leq i \leq K$ and $i \in N^+$ in the paper. It will become an increasing function at this time

$$\begin{aligned} t_{r2} &= \max\left(\cup_{i=1}^{K} t_e(i)\right) = t_e(i_3) \\ &= (K-1)\overline{t_d} + \left(\frac{S}{M} \text{ div } K\right) \cdot \overline{t_o} \end{aligned} \tag{7}$$

4.2.2 Multi-task rendering results upload time consumption

For t_{upload} part, since all upload tasks are carried out by the parallel way, this section is consistent with the single task system as show in the following formula,

$$t_{\text{upload}} = C\left(\frac{D}{W} + \Delta_w\right) + \frac{\Delta_d}{W} + \Delta_w + 2\Delta_{tcp} \tag{8}$$

In this paper, $T_{MR(A)}$ is defined as the time consumption of non-blocking status, $T_{MR(B)}$ is defined as the time consumption of the blocking status. Non-blocking task scheduling is a single master processor and there are worker/client processors. Each task has all the data it needs to compute, but gets the index to work on from the master. After the computation, the worker returns some data to the master. The bottom line is if a task takes too long to compute then it becomes the limiting factor and the master cannot move on to assign an index to the next worker even if non-blocking techniques are used. Is it possible to skip assigning to a worker and move on to next. $\mathrm{T}_{\mathrm{MR}(A)}$ can be expressed as

$$\mathrm{T}_{\mathrm{MR}(A)} = 4\left\lceil\frac{\mathrm{S}}{N}\right\rceil\left(\frac{\Delta_d}{W}+\Delta_w\right)+\mathrm{P}\Delta_p+\mathrm{C}\Delta_c + \mathrm{C}\left(\frac{\mathrm{D}}{W}+\Delta_w\right)+\Delta_w+\mathrm{Q}_1 \tag{9}$$

When the network is in good condition, $\Delta_w \approx 0$, so the formula can be further simplified as $\mathrm{T}_{MR(A)} = 4\lceil\frac{\mathrm{S}}{N}\rceil\frac{\Delta_d}{W}+(\mathrm{P}\Delta_p+\mathrm{C}\Delta_c)+\frac{\mathrm{CD}}{W}+\mathrm{Q}_1$.$\mathrm{T}_{\mathrm{MR}(B)}$ can be further simplified as $\mathrm{T}_{\mathrm{MR}(B)} = \mathrm{t}_0 + t_{r2} + t_{\mathrm{upload}} + \mathrm{t}_1$. It can achieve good communication when $\Delta_w \approx 0$, so we can obtain after the expansion and simplification,

$$\mathrm{T}_{\mathrm{MR}(B)} \approx \begin{cases} \frac{4M(\mathrm{K}-2)}{N}\cdot\frac{\Delta_d}{W}+\left(\frac{S}{M\cdot K}+1\right)(P\Delta_p+C\Delta_c) + \frac{\mathrm{CD}}{W}+\mathrm{Q}_1, & \left\lceil\frac{S}{M}\right\rceil \bmod K\neq 0 \\ \frac{4M(\mathrm{K}-1)}{N}\cdot\frac{\Delta_d}{W}+\frac{S}{M\cdot K}(P\Delta_p+C\Delta_c) + \frac{\mathrm{CD}}{W}+\mathrm{Q}_1, & \left\lceil\frac{S}{M}\right\rceil \bmod K = 0 \end{cases} \tag{10}$$

In conclusion, the multi-task rendering process time consumption T_{MR} can be summarized as

$$\mathrm{T}_{\mathrm{MR}}(S,N,M,K) = \begin{cases} \mathrm{T}_{\mathrm{MR}(A)}, K\overline{t_d} \geq \overline{t_o} \cup \mathrm{S} \leq \mathrm{M}\cdot\mathrm{K} \\ \mathrm{T}_{\mathrm{MR}(B)}, K\overline{t_d} < \overline{t_o} \cap \mathrm{S} > \mathrm{M}\cdot\mathrm{K} \end{cases} \tag{11}$$

Among them,

$$\overline{t_d} = \frac{4\mathrm{M}}{N}\left(\frac{\Delta_d}{W}+\Delta_w\right) \tag{12}$$

$$\overline{t_o} = P\Delta_\mathrm{p} + \mathrm{C}\Delta_\mathrm{c} \tag{13}$$

This paper draws the following conclusions by analysis of the main factors which affect the number of concurrent users and the average response delay : (1) the average response delay is affected by many factors: (1) the relationship of concurrent tasks number S and the multi-task rendering time T_{MR} is linear. (2) The relationship of web server number N and multi-task rendering time T_{MR} is inversely proportional. (3) The rendering time $P\Delta_\mathrm{p} + C\Delta_\mathrm{c}$ is the coefficient of parameter S in the blocking status. (4) The number of render machine M and the number of sites K will have a direct contribution to the T_{MR} in the blocking status. (2) The number of concurrent users is mainly restricted by the average response delay, because the increase of the number of concurrent users will directly lead to the increase of average response delay.

5 System performance optimization results and discussion

The performance pressure of the system mainly focuses on the multi-task rendering process. This paper will mainly analyze the optimal selection scheme of *K*, *M*, and *N* in the status of given *S* and *T*. Analysis process is divided into the following: (1) we analyze the performance degradation trends of different status under this process. (2) In the face of the specified *S* level, we can not only observe the trends of *N-T*, *M-T*, and *K-T*, but also can observe the change trend of *N-S*/*T*, *M-S*/*T*, and *K-S*/*T*, and ultimately choose a better *N*, *M*, *K* ratio by the directional derivative. (3) The derivation of discussion also contains scene rendering optimization, scheduling algorithm optimization, and expansion support.

5.1 The first test experiments

We conduct experiments on our own systems and models. On the implementation of the test process, four sets of test input parameters were used to test the performance of the six cloud rendering systems on the two machines. Above figure shows the running situation of some rendering programs on the rendering machine A. And each rendering program has a corresponding command window to display the current program running log information.

5.2 The second test experiments

In order to test the efficiency of the algorithm, we tested it on a publicly available large-scale simulation scenario (http://pointclouds.org/ [23]) provided by Middlebury, Canada. The data source is shown in Fig. 1,

These data source properties are shown in the following Table 1,

5.3 Status transition function

Assume function $G(N, M, K)$ expresses the relationship of $K\overline{t_d}$ and $K\overline{t_d}$. They are defined as $G(N,M,K) = K\overline{t_d} - \overline{t_o} = \frac{\mathrm{M}\cdot\mathrm{K}}{N}\cdot\frac{4\Delta_d}{W} - (P\Delta_p + C\Delta_c)$. It is not difficult to find that while $G(N,M,K) \geq 0$, the system will be in non-blocking status, while $G(N,M,K) < 0$ the system is in a blocked status.

The performance bottleneck analysis is shown in the following figure. It can be seen from the diagram, in the system, that the main time-consuming part is in two-dimensional image rendering, file upload, and file transfer.

5.4 Average response time analysis

The average response time is an important index in the system performance analysis, which directly determines the user experience. In the multi-task rendering process, the more common situation is the $\lceil\frac{S}{M}\rceil \bmod K\neq 0$; this

(1) Virtual park scene (2) Power plant scene (3) Virtual city scene

Fig. 1 The 3D scene test data source of performance bottleneck analysis and resource optimized distribution method. (1) Virtual park scene of the data source. (2) Power plant scene of the data source. (3) Virtual city scene of the data source. This group figure describe the test data source of performance bottleneck analysis and resource optimized distribution method for IoT cloud rendering computing system in cyber-enabled applications. These data source provided by Middlebury, Canada (http://pointclouds.org/ [23]). We select the test models with the largest number of patches, vertexes, and the largest amount of data as experimental test model data. These representative models are selected according to the standard of include most IoT cloud rendering computing system scenarios and most amount of data. Moreover, these models have typical loading times and operational characteristics. If our algorithm works well on these characteristics models, it will work well on other scene models as well. In this group of models and scenes, we selected some 3D model test data set from Middlebury, Canada. The first sub-figure is a virtual park scene and related models, the second sub-figure is a power plant scene and related models, and the third sub-figure is a larger virtual city scene and related models. The following are advantages of scene and models:(1) under different influence of network congestion, these scenes and models show different rendering effects. These models are greatly influenced by different network speed, network congestion, algorithm, and parameters. So this kind of model can distinguish different network congestion situations and different algorithms efficiency. It could truly distinguish the advantages and efficiency of different method. (2) These scenes and models contain more vertexes, edges, and triangular patches, which can distinguish complex method effect easily in graphics. (3) These scene and models include the brightness, hue, and saturation of scene color. These scenes and models will show different rendering effects under different rendering method and procedures. These scenes and models have high discrimination and expressiveness for different algorithms

paper choose $T_{\mathrm{MR}(B)}=\frac{4M(\mathrm{K}-2)}{N}\cdot\frac{\Delta_d}{W}+(\frac{S}{M\cdot K}+1)(P\Delta_p+C\Delta_c)+\frac{\mathrm{CD}}{W}+\mathrm{Q}_1$. $T_{\mathrm{MR}(B)}$ substract $T_{\mathrm{MR}(A)}$ can obtain the time difference $T_\Delta=T_{MR(B)}-T_{MR(A)}=\frac{4\Delta_d(M(\mathrm{K}-2)-\mathrm{S})}{N\cdot W}+\frac{S}{M\cdot K}(P\Delta_p+C\Delta_c)$. The average response time difference can be obtained by T_Δ divided by S, namely, $\overline{T_\Delta}=\frac{T_\Delta}{S}=\frac{4M\Delta_d(\mathrm{K}-2)}{N\cdot W\cdot S}+\frac{1}{M\cdot K}(P\Delta_p+C\Delta_c)-\frac{4\Delta_d}{N\cdot W}$. According to the status, transfer function shows that when $G(N, M, K) = 0$, the system at the critical points. So the status $G(N,M,K)=0\rightarrow\frac{\mathrm{M}\cdot\mathrm{K}}{N}\cdot\frac{4\Delta_d}{W}-(P\Delta_p+C\Delta_c)=0\rightarrow\frac{1}{M\cdot K}(P\Delta_p+C\Delta_c)-\frac{4\Delta_d}{N\cdot W}=0$. Because of $K\geq 2$, so $\frac{4M\Delta_d(K-2)}{N\cdot W\cdot S}\geq 0$ was established. So we can get the conclusion that when the system is in the critical points of the blocking and non-blocking status, the average response time $T_{\mathrm{MR}(A)}$ is better than $T_{\mathrm{MR}(B)}$.

5.5 System performance decline rate analysis

$T_{\mathrm{MR}}(S,N,M,K)$ can be used to represent the time consumed by the multi-task rendering process. S can be seen as the main variable function. Because the value of the variable with the number of users will change at any time. While the remaining variables can be regarded as the secondary variables, these variables will set the default values in the cloud rendering system T_{MR} calculate partial derivation for S. In order to facilitate the discussion, the following variables are further defined:

Table 1 Experimental data model attributes

Model name	Patch number	Vertex number
Virtual park scene	3,319,336	105,072
Vertex number	1,707,673	106,898
Frame number	12,035	142,379

$$T'_{S1}=\frac{4\Delta_d}{N\cdot W'}\;T'_{S2}=\frac{P\Delta_p+C\Delta_C}{M\cdot K} \tag{14}$$

T'_{s1} indicate that there is a performance bottleneck in task scheduling part of the system. T'_{s2} indicates that there is a performance bottleneck in the rendering part of the system. The T_{MR} in $(M\cdot K,+\infty)$ interval growth are discussed, while the conditions $\Delta_w=0$, the $G(N, M, K)<0$ are satisfied, and the T'_{s2},

$$T'_{s2}=\frac{P\Delta_p+C\Delta_c}{M\cdot K}=\frac{\overline{t_o}}{M\cdot K}>\frac{\overline{t_d}}{M}=\frac{4\Delta_d}{N\cdot W}=T'_{s1} \tag{15}$$

The results show how to adjust other parameters regardless. T_{MR} in $(M\cdot K,+\infty)$ status interval change will always tend to growth faster aspect. However, after determining the growth rate of T_{MR}, it still can adjust the parameters to improve the performance of the system. In order to test the algorithm of this paper, we choose the current popular algorithm [21–23] to test system pressure. After blocking, the average response time of the algorithm is shown in the following Table 2.

5.6 System performance tuning strategy

When $G(N,M,K)\geq 0$, the growth rate of T_{MR} is T'_{s1}, and there is an upper limit of N,

Table 2 The average response time of each test algorithm after blocking

3D scene	Average render response time (ms)		
	VFC algorithm [21]	CHC algorithm [22]	CHC++ algorithm [23]
Virtual park scene	44.5	50.24	39.63
Power plant scene	50.42	40.61	39.81
Virtual city scene	45.76	40.89	39.92
Average	46.89	43.38	39.78

$$\max(N) = \left\lfloor \frac{4MK\Delta_d}{W(P\Delta_p + C\Delta_c)} \right\rfloor \tag{16}$$

We can make the system degrade the growth rate of T'_{s1} to a minimum to maintain the current status of the system by adjusting the parameter N to achieve upper limit value. When $G(N, M, K) < 0$, the growth rate T_{MR} is T'_{s2}, we can reduce the growth rate by the following ways: increase the product size of the parameters $M \cdot K$ and reduce the size $P\Delta_p + C\Delta_c$. So we can choose to upgrade $M \cdot K$ to the upper limit

$$\max(M{\cdot}K) = \left\lfloor \frac{N{\cdot}W}{4\Delta_d}(P\Delta_p + C\Delta_c) \right\rfloor \tag{17}$$

The purpose is to make the $G(N, M, K) \to 0$. When $G(N, M, K) \to 0$, task scheduling part and rendering part have the same performance. When $N \to +\infty$, $\lim_{N\to+\infty} G(N,M,K) \to -(P\Delta_p + C\Delta_c)$, it means that the performance of rendering part is serious behind task scheduling parts, and the system operation is too slow so that becomes a performance bottleneck in the rendering part. When $\mathrm{M} \cdot \mathrm{K} \to +\infty$, $\lim_{N\to+\infty} G(N,M,K) \to +\infty$, it means that the performance of task scheduling part is serious behind the rendering part so that the system operation is too slow to become a performance bottleneck in task scheduling part. The average response time of each algorithm after system optimization is shown in the following Table 3,

Table 3 Average response time of each algorithm after system optimization

3D scene	Average render response time (ms)		
	VFC algorithm [21]	CHC algorithm [22]	CHC++ algorithm [23]
Virtual park scene	14.5	10.24	9.63
Power plant scene	20.42	10.61	9.81
Virtual city scene	15.76	10.89	9.92
Average	16.89	10.58	9.78

6 Conclusion

At present, the cloud computing system and cloud rendering industry are substantially rising. Aiming at the current cloud rendering system performance, we propose a set of diagnostic performance bottlenecks and resources to optimize allocation methods and to analyze the core performance of cloud rendering system by this theory, especially to analyze the multi-task rendering process. We can obtain the following conclusions: (1) the average response delay is influenced by many factors. (2) The increase of concurrent users' number will directly lead to the increase of average response delay. (3) The parameters parts of concurrent users' number will affect the concurrent user number. We propose a unique parameter adjustment strategy to improve system performance by rigorous mathematical proof, namely, under $G(N, M, K) \geq 0$ circumstances, we optimize the system by maximizing the parameter N to the upper limit, or in $G(N, M, K) < 0$ circumstances, we optimize the system by increasing the parameter $M \cdot K$ product to limit the decrease rate of T_{MR}. This is a new performance optimization scheme for cloud rendering system.

Abbreviations

ACM: Adaptive coded modulation; AWGN: Additive white Gaussian noise; CHC: Coherent hierarchical culling; CHC++: Coherent hierarchical culling revisited; VFC: View frustum culling

Acknowledgements

We sincerely thank each one of the reviewer and editors' work to this paper. This paper is supported by National Key R&D Program of China 2017YFC0821603, Beijing NOVA Program of China (Grant No. Z181100006218041). The National Key Research and Development Program of China: Research and development of intelligent security card port monitoring and warning platform(Grant No. 2016YFC0800507), Innovation Foundation Program of China Electronics Technology Group Corporation: Research on holographic and abnormal behavior intelligent warning technology for social security risk targets.

Funding

The research presented in this paper was supported by the Beijing NOVA Program of China, Ministry of science and technology of China.

Availability of data and materials

The data and materials in this paper are all true and available.

Authors' contributions

RHW is the main writer of this paper. He proposed the main idea of the algorithm, joined the whole experiment, and calculated the results. BZ optimized the parameters of the method. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Author details
[1]National Engineering Laboratory for Public Security Risk Perception and Control by Big Data (PSRPC), China Academy of Electronics and Information Technology, Beijing, China. [2]State Key Laboratory of Software Developing Environment, School of Computer Science and Engineering, Beihang University, Beijing, China. [3]National Engineering Laboratory for Big Data Application Technologies for Comprehensive Traffic, Beijing, China.

Received: 7 October 2018 Accepted: 14 March 2019
Published online: 29 March 2019

References

1. S. Wang, S. Dey, Cloud mobile gaming: modeling and measuring user experience in mobile wireless networks. ACM SIGMOBILE Mob. Comput. Commun. Rev. **16**(1), 10–21 (2012)
2. Z. Zhao, K. Hwang, J. Villeta, in *Proceedings of the 3rd ACM workshop on Scientific Cloud Computing Date*. Game cloud design with virtualized CPU/GPU servers and initial performance results (2012), pp. 23–30
3. N. Tizon, C. Moreno, M. Cernea, et al., in *Proceedings of the 16th ACM International Conference on 3D Web Technology*. MPEG-4-based adaptive remote rendering for video games (2011), pp. 45–50
4. R. Wang, B. Zhang, J. Bi, et al., *Multimed Tools Appl* (2018). https://doi.org/10.1007/s11042-018-6569-1
5. S. Shi, M. Kamali, K. Nahrstedt, et al., in *Proceedings of the 18th ACM international conference on Multimedia*. A high-quality low-delay remote rendering system for 3D video (2010), pp. 601–610
6. W. Wu, A. Arefin, G. Kurillo, et al., in *Proceedings of the 19th ACM international conference on Multimedia*. Color-plus-depth level-of-detail in 3D tele-immersive video: a psychophysical approach (2011), pp. 13–22
7. W. Wu, A. Arefin, G. Kurillo, et al., CZLoD: A psychophysical approach for 3D tele-immersive video. ACM Trans. Multimed Comput. Commun. Appl. (TOMCCAP) **8**(3s), 39 (2012)
8. S. Shi, C.H. Hsu, K. Nahrstedt, et al., in *Proceedings of the 19th ACM international conference on Multimedia*. Using graphics rendering contexts to enhance the real-time video coding for mobile cloud gaming (2011), pp. 103–112
9. S. Shi, K. Nahrstedt, R. Campbell, A real-time remote rendering system for interactive mobile graphics. ACM Trans. Multimed. Comput. Commun. Appl. (TOMCCAP) **8**(3s), 46 (2012)
10. P. Ndjiki-Nya, M. Koppel, D. Doshkov, et al., Depth image-based rendering with advanced texture synthesis for 3-D video. IEEE Trans. Multimed. **13**(3), 453–465 (2011)
11. M. Koppel, X. Wang, D. Doshkov, et al., in *Proceedings of the 19th IEEE International Conference on Image Processing (ICIP 12)*. Depth image-based rendering with spatio-temporally consistent texture synthesis for 3-D video with global motion (2012), pp. 2713–2716
12. M. Zhu, S. Mondet, G. Morin, et al., in *Proceedings of the 19th ACM international conference on Multimedia*. Towards peer-assisted rendering in networked virtual environments (2011), pp. 183–192
13. Pajak D, Herzog R, Eisemann E, et al. Scalable Remote Rendering with Depth and Motion-flow Augmented Streaming. Proceedings of the 32th annual conference of the European Association for Computer Graphics (EG 04), 2011, 30(2): 415–424
14. M. Zamarin, P. Zanuttigh, S. Milani, et al., in *Proceedings of the ACM International Workshop on 3D Video Processing*. A joint multi-view plus depth image coding scheme based on 3D-warping (2010), pp. 7–12
15. Y. Liu, Q. Huang, S. Ma, et al., A novel rate control technique for multiview video plus depth based 3D video coding. IEEE Trans. Broadcast. **57**(2), 562–571 (2011)
16. T. Süß, C. Koch, C. Jähn, et al., in *Proceedings of Graphics Interface*. Approximative occlusion culling using the hull tree (2011), pp. 79–86
17. J. Liu, N. Zheng, L. Xiong, et al., *Illumination transition image: parameter-based illumination estimation and re-rendering* (2008), pp. 1–4
18. Huang Yu, Zhang Chao, A layered method of visibility resolving in depth image-based rendering[C]// 19th International Conference on Pattern Recognition, ICPR 2008, Institute of Electrical and Electronics Engineers Inc., Tampa Convention Center Tampa (Florida, DBLP, 2008), pp. 1-4
19. R.M. Cadena, R. De La Cruz, E. Sergio Bayro-Corrochano, Rendering of brain tumors using endoneurosonography[C], 19th International Conference on Pattern Recognition, ICPR 2008, Tampa Convention Center Tampa (Florida, IEEE Computer Society, 2008), pp. 1-4.
20. K. Zeng, M. Zhao, C. Xiong, et al. From image parsing to painterly rendering[J]. Acm. Trans. Graph. Assoc. Comput. Machinery. **29**(1),1-11 (2009)
21. L. Ballan, G.J. Brostow, J. Puwein, M. Pollefeys, Unstructured video-based rendering: interactive exploration of casually captured videos[J]. Acm. Trans. Graph. Assoc. Comput. Machinery. **29**(4), 157-166 (2010)
22. E. Wood, T. Baltruaitis, X. Zhang, Y. Sugano, P. Robinson, A. Bulling, rendering of eyes for eye-shape registration and gaze estimation[C], 2015 IEEE International Conference on Computer Vision (ICCV) (IEEE Computer Society, Santiago, 2015), pp. 3756-3764
23. S. Pujades, F. Devernay, B. Goldluecke, Bayesian view synthesis and image-based rendering principles[C], IEEE Computer Society Conference on Computer Vision and Pattern Recognition (IEEE Computer Society, Columbus, 2014), pp. 3906-3913
24. Xiong Ying, Saenko Kate, Darrell Trevor, Zickler Todd, From pixels to physics: Probabilistic color de-rendering[C], IEEE Computer Society, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2012, Providence, 2012. pp. 358-365
25. O. Aldrian, W.A.P. Smith, Inverse rendering of faces on a cloudy day [M]// Computer Vision – ECCV 2012. (Springer Verlag, Florence, 2012). pp. 201-214.
26. C. Lei, X.D. Chen, Y.H. Yang, A new multi-view space time-consistent depth recovery framework for free viewpoint video rendering[C]// IEEE Computer Society, International Conference on Computer Vision. (ICCV, Kyoto, 2009). pp. 1570-1577.
27. T.C. Bler, T. Rittig, E. Kasneci, et al. Rendering refraction and reflection of eyeglasses for synthetic eye tracker images[C]// Biennial ACM Symposium on Eye Tracking Research & Applications. (Association for Computing Machinery, Charleston, 2015). pp. 143-146.
28. R.O. Cayon, A. Djelouah, G. Drettakis, A Bayesian approach for selective image-based rendering using superpixels[C]// 2015 International Conference on 3D Vision, 3DV 2015. (Institute of Electrical and Electronics Engineers Inc, Lyon, 2015). pp. 469-477
29. L. Świrski, N. Dodgson, in *Symposium on Eye Tracking Research and Applications*. Rendering synthetic ground truth images for eye tracker evaluation (2014), pp. 219–222
30. O. Aldrian, W.A.P. Smith, *Inverse rendering with a morphable model: a multilinear approach* (2011)
31. G. Cheung, V. Velisavljevic, A. Ortega, On dependent bit allocation for multiview image coding with depth-image-based rendering. IEEE Trans. Image Process. Publication IEEE Sig. Process. Soc. **20**(11), 3179–3194 (2011)
32. G. Petrazzuoli, M. Cagnazzo, F. Dufaux, et al, Using distributed source coding and depth image based rendering to improve interactive multiview video access[C]// IEEE International Conference on Image Processing. (IEEE Computer Society, ICIP, Brussels, 2011). pp. 597-600
33. M. Ishii, K. Takahashi, T. Naemura, Joint rendering and segmentation of free-viewpoint video. J Image Video Process **2010**(1), 3 (2010)
34. S. Smirnov, A. Gotchev. Real-time depth image-based rendering with layered dis-occlusion compensation and aliasing-free composition[C]// SPIE/IS&T Electronic Imaging. (International Society for Optics and Photonics, SPIE, San Francisco, 2015). pp. v93990T-93990T-11.
35. Han Mahn-jin 307–602 Satbyeol Maeul Woobang Apt, Ignatenko A. Image-based method of representation and rendering of three-dimensional object: EP, EP1271410[P]. 2010
36. S.K. Chow, K.L. Chan, Fast and realistic rendering of deformable virtual characters using impostor and stencil buffer. Int. J. Image Graph. **06**(4), 599–624 (2011)
37. M. Xi, L.H. Wang, Q.Q. Yang, et al., Depth-image-based rendering with spatial and temporal texture synthesis for 3DTV. Eurasip J. Image Video Process. **2013**(1), 9 (2013)
38. L.M. Po, S. Zhang, X. Xu, et al, A new multidirectional extrapolation hole-filling method for Depth-Image-Based Rendering[C]// IEEE International Conference on Image Processing, ICIP 2011. (IEEE Computer Society, DBLP, Brussels, 2011). pp. 2589-2592
39. H.W. Cho, S.W. Chung, M.K. Song, et al., Depth-image-based 3D rendering with edge dependent preprocessing. Midwest Symp. Circuits Syst. **47**(10), 1–4 (2011)
40. W.L. Gaddy, V. Seran, Y. Liu, *System and method for transmission, processing, and rendering of stereoscopic and multi-view images: US, US8774267[P]* (2014)
41. M. Solh, G. Alregib. Hierarchical Hole-Filling(HHF): Depth image based rendering without depth map filtering for 3D-TV[C]// IEEE International Workshop on Multimedia Signal Processing. (IEEE Computer Society, IEEE Xplore, Saint Malo, 2010). pp. 87-92

42. C. Cigla, A.A. Alatan, An efficient hole filling for depth image based rendering[C]// IEEE International Conference on Multimedia and Expo Workshops. (IEEE Computer Society, San Jose, 2013). pp. 1-6
43. C.D. Herrera, J.H. Kannala, et al, Multi-view alpha matte for free viewpoint rendering[C]// Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), v 6930 LNCS, p 98-109, 2011. Rocquencourt, France, Springer-Verlag, 2011:98-109
44. C. Liu, Z. Lai, M. Dong, J. Hua, Multi-instance rendering based on dynamic differential surface propagation[C]// IEEE International Conference on Image Processing. (ICIP, IEEE Computer Society, Lake Buena Vista, 2012). pp. 3005-3008
45. H. Huang, T.N. Fu, C.F. Li, Painterly rendering with content-dependent natural paint strokes. Vis. Comput. **27**(9), 861–871 (2011)
46. Y. Zang, H. Huang, C.F. Li, Artistic preprocessing for painterly rendering and image stylization. Vis. Comput. **30**(9), 969–979 (2014)
47. X. Ning, H. Laga, S. Saito, et al., Contour-driven Sumi-e rendering of real photos. Comput. Graph. **35**(1), 122–134 (2011)

Submit your manuscript to a SpringerOpen journal and benefit from:

► Convenient online submission
► Rigorous peer review
► Open access: articles freely available online
► High visibility within the field
► Retaining the copyright to your article

Submit your next manuscript at ► springeropen.com