Object Detection and Parallel Rendering OpenGL-based Graphics Applications in Mobile Cloud Computing
Cong-Thinh Huynh, Tien-Dung Nguyen, Mauricio Gómez and Eui-Nam Huh
Department of Computer Engineering, Kyung Hee University

Abstract
Recently, together with the ongoing development of mobile device, mobile applications and cloud computing, mobile cloud computing (MCC) is soon to be a potential technology for mobile services, where mobile application are built, powered, deployed and hosted on cloud computing infrastructure. This paper describes object detection and parallel rendering architecture on cloud computing servers by using the power of cloud computing to detect an object in 3D application, delivery it to compute cluster servers and transfer to mobile client devices.

1. Introduction
Nowadays, OpenGL is the most widely used as the premier environment for developing portable, interactive 2D, 3D graphics application [1], bringing thousands of applications to a wide variety of computer, mobile devices [2], especially in offline, online games for computer and mobile. Despite the recent development of hardware technology, many 3D graphics applications which deployed on cloud computing server still cannot run at acceptable speed and graphics quality on mobile client. The causes are not only due to the network infrastructure, hardware specifications, rendering algorithm, processing and computing technique but also the limitation of mobile devices such as CPU, memory, storage, battery, etc.; the 3D graphics applications cannot perform efficiently. Therefore, the needs for advanced object detection and rendering technique are proposed.

Our work is focus on the server side, where the application is deployed. Fig. 1 demonstrates an OpenGL-based 3D games with many objects, based on the OpenGL commands, we can detect the objects in application source code such as a warrior, a person, a tree and intercept the OpenGL commands to rendering server to render these objects then delivery to rendered images results to mobile thin client. The goal of this paper is propose new novel ideas about object detection based on application source code and improve the stream server by design a new rendering technique called parallel rendering.

The rest of this paper is organized as follows. Section 2 presents the Related Works, section 3 presents Object Detection technique. In the section 4, we describe the Parallel rendering algorithm. We finally show the results in section 5 and conclusions in section 6.

Figure 1. 3D graphics games with many objects

2. Related Works
In the area of object detection and rendering, X and GLX windows are two most widely used solutions. GLX [4] is a wire protocol for OpenGL that allows an application to transmit a simple packed representation
of the OpenGL command parameters to a server and then to be executed on behalf of the client [3].

In the area of parallel geometry processing, Torborg [5] presents a solution whereby state commands are immediately broadcast to each geometry processor.

3. Object Detection based on source code

In an OpenGL-based 2D/3D graphics application, each object is drawn, manipulated, rendered by OpenGL commands. Those objects have their own states, where the color, shape, light, position, movement, etc. are defined, therefore the whole application will be a large state set of OpenGL commands. The state commands cannot be packed and bucketed immediately, because of the geometric extent and they may affect future geometry when a new object status, position, movement are changed and when the object states are detected, we can intercept, destine them to many different rendering server.

4. Parallel Rendering

By using the parallel technique, we can achieve a large scale application computational based on the power of workstation hardware. Fig. 3 describes how to compute an application process using single-thread and the more advantage method with multi-thread.

And the process can be processed as below steps.

//Thread_1:
wglMakeCurrent(NULL, NULL);
waitForThread2(); OrDoSomethingElse();
wglMakeCurrent(dc, glrc);

//Thread_2:
wglMakeCurrent(dc, glrc);
DoSome_OpenGL_Commands();
wglMakeCurrent(NULL, NULL);
Terminate_Thread2_And_GiveControlToThread1();

Figure 2. Object state tracking

Figure 3. Multi-thread computing

Figure 4. Multi-thread computing
By detecting object states and dividing them into group of application nodes, where each node represents an object state set. Then, the system delivery those application nodes to parallel rendering servers and transfer the rendered outputs (partial image) to device screen as shown in Fig. 5.

5. Results
By using this technique, we can detect the object and its state, movement, position, color, shape, etc. and transfer to rendering servers to render each object then delivery to mobile devices. Fig. 6 represents how system works with two game characters rendered on cloud servers after rendering process.

Figure 6. Object detection and rendering on servers

6. Conclusion
In this paper, we have proposed the object detection and parallel technique to detect the objects, object states based on OpenGL source code and using the power of cloud computing server resources to render multi-object to get the final rendered images. This technique also can be used not only on cloud server for games server but also for various large scale 2D/3D graphics applications.

Acknowledgement
This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (2010-0020725). Professor Eui-Nam Huh is corresponding author.

References