MOBILE APPLICATION DEVELOPMENT IN A COLLABORATIVE AEC DESIGNING SYSTEM

Zhuohua Huang, Engineer, huangzhuohua@cabrtech.com
Yang Liu, Engineer, clxy@vip.qq.com
Encheng Ma, Research Fellow, maencheng@cabrtech.com
Institute of Building Engineering Software, CABR Tech Co., LTD, Beijing,, China

ABSTRACT

The mobile devices act as a more and more important role in people’s daily life and the internet technology has been increasing at an astonishing speed; it is significant to introduce more mobile applications into AEC industry in order to build an efficient way of communicating between owners, designers, constructors and other persons. This paper presents a mobile application which is designed to view architecture model, structure model or MEP model in 3D format; to view project information and documentation; to send feedback in text, image, sound or video format. The system is designed to have the capability to communicate among the engineers throughout the collaborative design platform in real-time. While the transmission speed of the large-scale 3D-models in a limited band width is the major problem to be solved to satisfy the practical application requirements. Based on the authors’ studies and development experience, the solutions on the algorithms of model file optimization and the vertex buffer object optimizations; a 3D picking algorithm based on AABB cubing; an area segmentation technique to decreases the complexity of a 3D object representations; and Irrlicht engine for the platform independent development have been obtained. A building model displayed on a mobile device system developed by the authors on the Android platform indicates that the solutions presented in this paper are effective.

Keywords: collaboration, BIM, real-time, rendering, mobile, smartphone, VBO, collision detection, octree

1. INTRODUCTION

In the last decades, the mobile devices had an astonishing growth rate and the smart phones became more and more popular throughout the world. It is possible to build a mobile solution to extend the collaborative AEC design system. The smart phones and tablet PCs are easy to carry and use. They could be brought into construction yard, vehicles, planes or many other places. It means that they can greatly increase the possible working locations for engineers. And the performance of the mobile devices is not far away from traditional personal computers, a 4-core CPU is a standard configuration for the latest Smartphone. In China, the numbers of the Smartphone users were more than 290million in 2012. [1]
It becomes very important to develop mobile applications for the AEC design system.

This paper presents a mobile application which is designed to view architecture model, structure model or MEP model in 3D format; to view project information and documentation;

to send feedback in text, image, sound or video format. The system is designed to have the capability to communicate among the engineers throughout the collaborative design platform in real-time. While the transmission speed of the large-scale 3D-model in a limited bandwidth is the major problem to be solved to satisfy the practical application requirements. Based on the authors' studying and development practice, the solutions were obtained which are presented in the following sections. A prototype is developed based on the technical solutions on the Android platform.

2. THE COLLABORATIVE AEC DESIGN SYSTEM

The Collaborative AEC Designing System is a distributed multi-user designing system that allows different users at different places to work together on the same building project at the same time. It allows users to access the building project through the traditional PC client program, web pages or mobile devices, the users can browse the components that the other users are working on and notify the others if there is anything wrong. The data server stores all project data and all user information; it gives data access to the clients and ensures that only the user with the right privilege can get the project data. And the workflow function is also given to keep the project processing in a planned procedure.

Figure 1 shows the architecture of a collaborative design system in which the mobile application acts as a package for the AEC engineers.

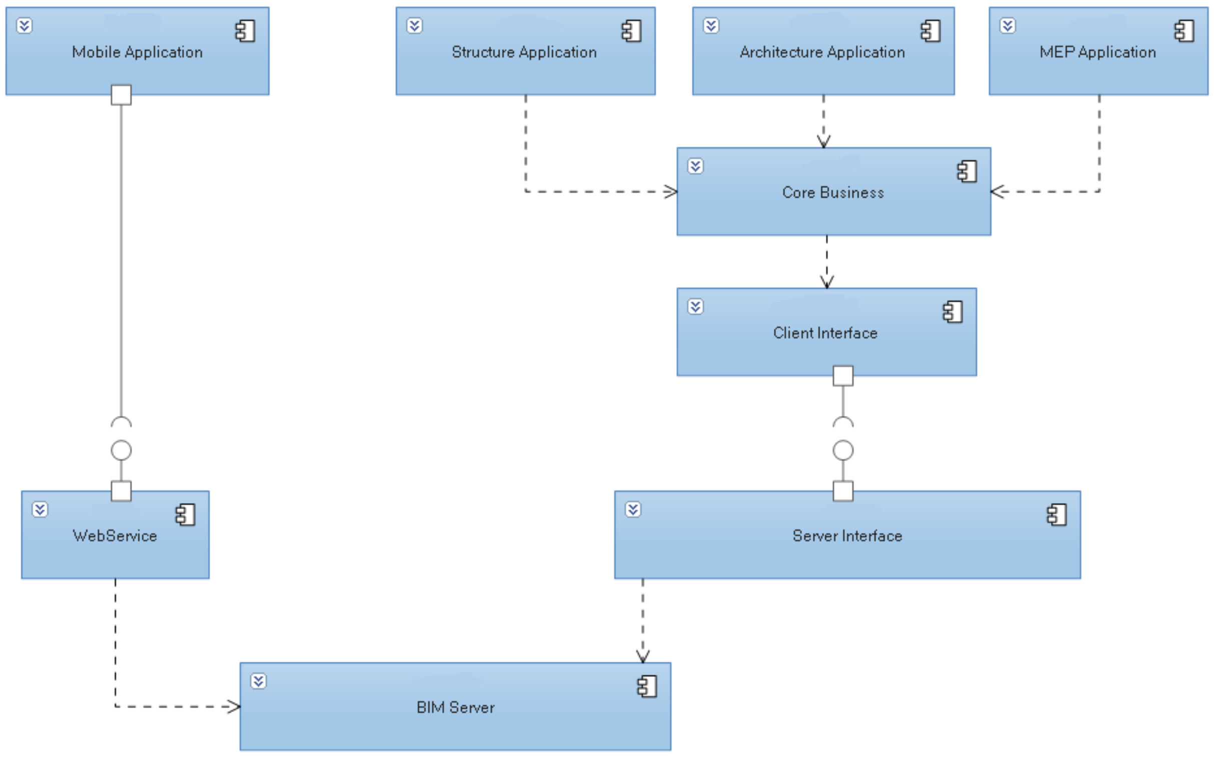

Figure 1: The architecture of a collaborative design system

3. MOBILE APPLICATION DEVELOPMENTS IN THE COLLABORATIVE SYSTEM

Currently there are very few companies have their mobile solutions for collaborative AEC design system. Through the BIM technology, more and more detailed information will be gathered and expected to be viewed or managed. Although the mobile devices still do not have the ability to run large-scale designing software, they can be used to browse the model and correct the wrong items in a design project. Based on the analysis of the mobile application requirements, the authors designed the following key features for a mobile application system:

3.1 3D model browsing

The mobile solution allows users to browse building models anywhere and anytime. It is built on mobile devices and can be run on Android, IOS, WP or other platforms.

Any user with the registered account of the collaborative AEC Designing System is allowed to log on through the mobile device and browse the building models. Roaming feature is provided that the user's viewpoint can be pushed forward or backward, indoor or outdoor, and auto roaming feature is made for the user to browse the building in a preset way. Each component of the building can be selected and all of its properties can be shown in the property window.

3.2 Message, voice and video feedback

Currently almost all smart mobile devices have voice, picture and video recording functions. This can be used in the mobile solution to improve the communication abilities for AEC projects.

Especially in the building site, an engineer can take a photo or video of the building under construction and bind them with a component of the model, then send the whole thing back. The designer will receive the feedback and notice where the failure takes place. If the designer makes some change on the model, it can also send the change set of component to mobile devices and the field engineer can check them again.

3.3 Project management

The most valuable function of a smart mobile device is allowing users to make use of fragments of time. The system is designed as follows: wherever there is a Wi-Fi Hotpoint or GPRS available, the manager can keep track of the project; file management functions allow the managers to view, delete, move, duplicate or rename all files on the server which he has the privilege; version management functions allow the manager to check any history of the project or roll the whole project or part of the project back to a specific version; workflow management functions allow the managers to set a workflow for the project, modify the current workflow or view the workflow processing statement; user management functions allow the managers to add, remove, view or modify the user's information, manage roles and user groups and change user privileges.

4. TECHNICAL SOLUTIONS

4.1 3D model data from mobile devices

In a Collaborative AEC Designing System, a building model may contain thousands of components and connections (The building model shown in Figure 2 contains 3000 components) while the transmission speed of the large-scale 3D-models in a limited band width is a big problem. It is important to design a proper and simple model data format to reduce the model so that the transmission speed of the large-scale 3D-models will not be affected by the limited band width. Obj file, X file and 3ds file are three possible options.

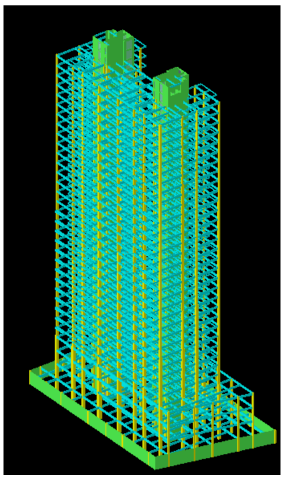
Figure 2: The model of a building

Obj file is a standard 3D model file format for 3D model and animation software "Advanced Visualizer" workstation version which is designed by Alias|Wavefront Co., Ltd .[2]. It stores all the data in text format and it could be opened and edited with notepad. Furthermore, it is supported by most of the well-known 3D software for its clarity and convenience. Obj file uses a stationary format for all kinds of data. For example, data starts with "v" stands for the coordinate of the vertex; data starts with "f" stands for triangle surface. We find that to use text format is intuitive and precise, but its shortcome is that the text format will use a lot of storage space.

X file is Direct3D platform specific file format; it is also stored in text format. For storage size issues, there is another storage mode for x file – binary mode. X file uses nesting data format to store coordinate, index, vector, texture and other data under different categories. Compare with Obj file, X file contents more information like initial matrix, initial configuration, and it has also more extendibility and abundance.

3ds file is based on binary block storage. It uses binary block to describe scenes data. It uses identification and offset value to record model data and its data block can be stored independently or stored as a nesting structure. By using binary block, 3ds file has a smaller file size than other text files and a faster data rebuild speed; but for using unreadable binary code, the storing process is complicated.

Based on the analysis of the above file formats, the authors learned that it is better to use text format to store model data in order to get an easier way for modification and verification, especially in the prototype phase of the project. In the future development, binary file format will be used to reduce the volumn of a building model so that the transmission speed in a limited band width will be increased. While the existed x file format and Obj file format do not meet the AEC application requirements well, so that a special model file format is designed to meet the requirements.

The following solutions are taken into account in designing the file format:

(1) OpenGL_ES hardware interface on mobile devices only supports triangulate polygon drawing. To get the interoperability between multiple AEC designing specifications, the triangulate surface data and the vertex information should be provided. And the other data like color, vector or texture should also be provided, so that the better visualization effects that close to real building can be obtained.

(2) To draw polygons, the vector index is required to be calculated and stored in the model file. For example, a cuboids model has 6*2*3 = 36 vertexes; and most of those vertexes are duplicated. If those vertexes are all stored in the model, more than 66% space will be wasted. Using vertex-index algorithm, 3 vertexes coordinate of each triangle will be gathered and the duplicated vertexes will be moved; therefore a lot of space can be saved. [3]

The minimal display unit of a building model is a component, and a component consists of several triangular surfaces with properties. To make sure that each of the components is unique and is easy to be searched by every client and the project server, the global id is introduced. For display performance requirements, the model file is optimized; the unnecessary properties and details is moved under the circumstance while the overall model and shape are not changed.

In this way, when reading the model, the only thing to do is to get the data in a loop using the file format and transfer them to the graphics engine as shown in Figure 3.

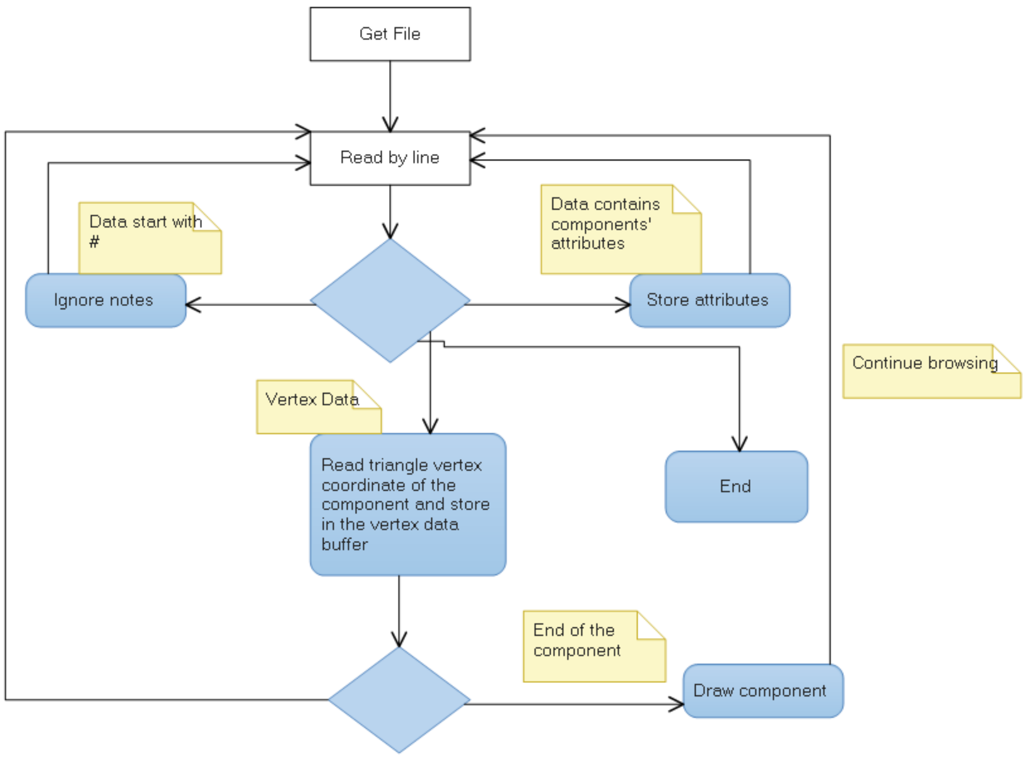

Figure 3: The flow chart to transfer the model to the graphics engine

4.2 VBO (Vertex Buffer Object) optimization

Although the model is optimized， some models of large and complex buildings may still be a heavy burden for mobile device. Due to this reason, a further optimization should be done.

The optimization algorithm is designed as: Minimizing the polygon redrawing counts when refreshing; storing the model vertex data which are often transferred from CPU to GPU into GPU memories in a temporary area.

Traditional 3D model drawing method use vertex and index data in the memory and send them to a GPU for calculation and rendering. Since this method is not efficient, VBO is designed to solve the problem. We create a buffer area in GPU to store vertex data and the data will only be written when they are initialized or changed. Using vertex buffer can significantly save the CPU-GPU copying cost and improve the program running efficiency.

It is important that VBO technology can only be used in hardware platform that based on OpenGL, the following functions like glGenBuffers, glBIndBuffer, and glBufferData are all OpenGL specific functions and they are mainly used for initialization of buffer data binding.

The methods for using VBO are as follows:

1) Use glGenBuffers function to distribute object number;

2) Use glBinderBuffer function of the bind buffer object, use GL_ARRAY_BUFFER to bind a vertex buffer or user GL_ELEMENT_ARRAY_BUFFER to bind index buffer;

3) Use glBufferData to bind buffer data, vertex buffer data and index buffer data will both be needed.

Figure 4 shows a model with 219,483 vertexes, 435,667 triangles. It takes 3*40,000 = 120,000 times for just one frame to render such a model using traditional glVertex function, To reach 30FPS, the function calls per second will be 3*40,000*30 = 36million, it will take a significant amount of time.

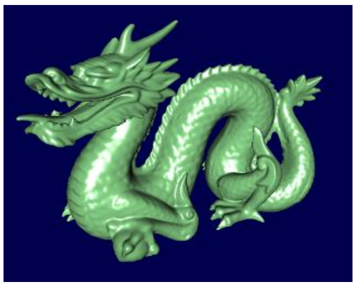

Figure 4: model using VBO rendering technology

An experiment is made to test the performance. The testing computer has an Intel Core2 6600 CPU, 2 Gigabytes memory and NVIDIA GeForce8600GT graphics card. The result is shown in Figure 5: the y axis stands for the frames and the x axis stands for different rendering methods. Using glVertex function can only get 1 FPS, far under the real time rendering requirements; Using Display list will get 16 FPS, but still not enough for the real time rendering. Finally, using VBO function in the same condition will end up with 60FPS, that is almost 4 times faster than Display list function; and it is adequate for real time rendering for AEC projects.

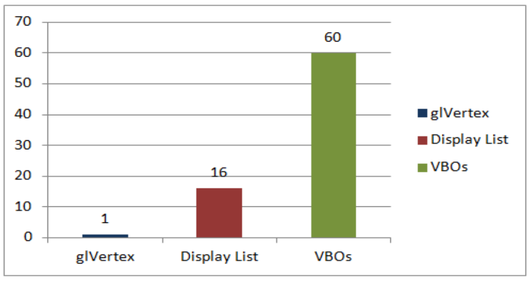

Figure 5" Efficiency comparison using different rendering methods

VBO technology resolves the transferring problem between CPU and GPU; greatly use the parallel calculation ability of the GPU. According to another experiment that was done in the Android platform with a 4-core 1.4G Hz CPU, Mali-400mp4 graphics card, 30 FPS can be reached when running a model with 150,000 triangles.

4.3 Collision detection algorithm

Viewing of component details is frequently required by the AEC users when they are browsing a building model. The collision detection algorithms should be carefully designed since the computer storage space and transmission speed is limited for the current available mobile devices. The 3D ray picking technology is used to get better user experience when the user is roaming the scene. To keep the viewpoint away from getting through the wall, a set of collision detection functions is built to support roaming and 3d picking. For mobile devices, the CPU and GPU speed is limited. An efficient collision detection algorithm should be made to ensure the browsing effect.

4.3.1 3D picking algorithm based on axis-aligned bounding box

The Axis-Aligned Bounding Box(AABB) is a cube that bound the object, it is used to simplify the collision detection. Figure 6 illustrates a 3D picking solution. The mobile device gets the user's long push signal and converts it to a 2-D coordinate, and then calculates its 3D coordinate in the camera axis and also the absolute coordinate in the virtual axis system. Throughout those two coordinates we can calculate the analytical equation of the line, and then together with the virtual AABB out of the component, the object to be selected can then be calculated.[4]

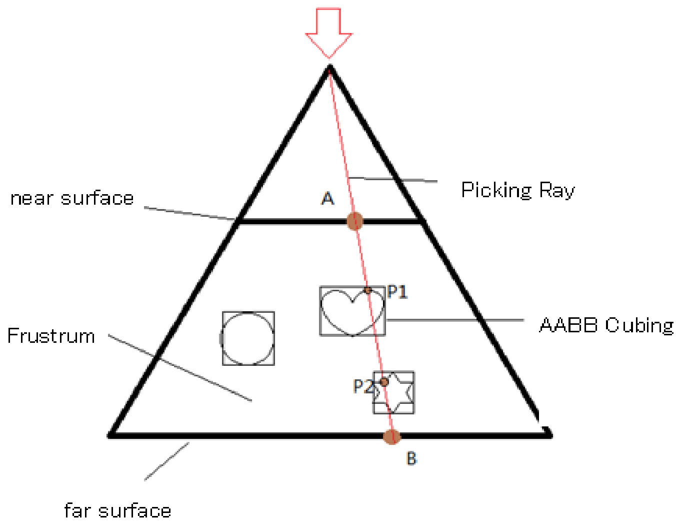

Figure 6 3D picking

The picking algorithm is presented in the following program codes:

```
//calculate the touched point coordinate
    float x0=x-w/2;
    float y0=h/2-y;
//Calculate the x axis, y axis coordinates in the near surface
    float xNear=2*x0*left/w;
    float yNear=2*y0*top/h;
// Calculate the x axis, y axis coordinate in the far surface
    float ratio=far/near;
    float xFar=ratio*xNear;
    float yFar=ratio*yNear;
//Calculate the absolute coordinate throughout the camera coordinate
    float[] A = MatrixState.fromPtoPreP(new float[] { xNear, yNear, near });
    float[] B = MatrixState.fromPtoPreP(new float[] { xFar, yFar, far });
```

After getting the absolute coordinates of two points, the AABB vertex coordinates can be iterated which indicates that the line of those two points have crossover points with AABB. The object that the user wants to select is therefore found.

The defection of the AABB algorithm is the deviation which cannot be ignored when calculating the irregular object. Another algorithm is needed in that case, so that the AABB counts can be reduced when detecting the collision.[5]

4.3.2 Octree technology based on spatial decomposition method

Octree is a spatial decomposition data structure. The Octree algorithm is to divide the whole virtual 3D space into eight equal units, then execute the process recursively and build the actor model(As shown in Figure 7). Octree method transfers the object collision problem into their points' collision problem. The method is extremely efficient when the object counts are limited and are evenly spreading in the space.

For a 3D model, most components are far away from each other and they are even in the whole building. It is to say that using Octree for viewpoint collision detection in the building 3D model is reasonable and efficient. To get higher rendering speed, camera clipping technique can also be used to save rendering cost through AABB fast clipping method. [6]

If the scene to render is complex, the octree collision detection and camera clipping will not be fast enough. The solution is using area segmentation technique and providing a partial model in different precision by the distance (Lod technique). It decreases the complexity of a 3D object representation as it moves away from the viewer so that the rendering speed and the visual effects are both satisfied.

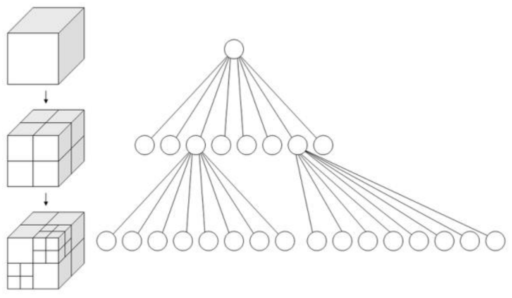

Figure 7: Octree dividing method

4.4 Cross-platform issues

To low down the development cost, the graphics engine should fit for multiple mobile platforms such as IOS, Android, WP, Blackberry, etc.. Several solutions have been studied, for example, QT, Rhodes, Mono, etc.; but they are not quite fit for our requirements. The final decision is made on using native language to develop a 3D core engine and use original platform language for another part of the solution.

To develop a 3D graphics engine from the very beginning is time consuming. It is more reasonable to find an open-source solution and do some modification and transplantation on it.

Irrlicht engine is a high performance real-time 3D graphics engine written by C++, it can be used in C++ or .Net project. Using Direct3D, OpenGL 1.2 or its own rendering program, Irrlicht can be platform independent. It also provides commercial-grade features such as dynamic shade, particle system, role animation, indoor and outdoor technic, collision detection, etc..

The original Irrlicht engine framework is designed to use Interface&Implementaion method, all its functions are under the Irr namespace. To be the 3D-graphic solution of our project, the code must be refactored.

First of all, the model file got from the collaborative server cannot be used directly. A specific interface should be developed to fulfill the requirements of the simplified building model and then the model data can be transferred into the interface by the array.

The Dataservice class is the model-data processing class, it calls nativeOnBuildingWorld function to transfer ordered model data to 3D engine. And JNI technic is used to pass the data between Java programs and C++ program.

Another issue is about gesture capturing, the capturing method is supported by the original system's API provided by each platform, thus this program should be implemented with the original language of each platform.

There are still some other issues in using Irrlicht engine, but the core functions are already implemented in Android platform and IOS platform.

Based on the solutions presented in this paper, the authors developed a mobile application prototype. Figure 8 shows a building model in a mobile device displayed by this prototype.

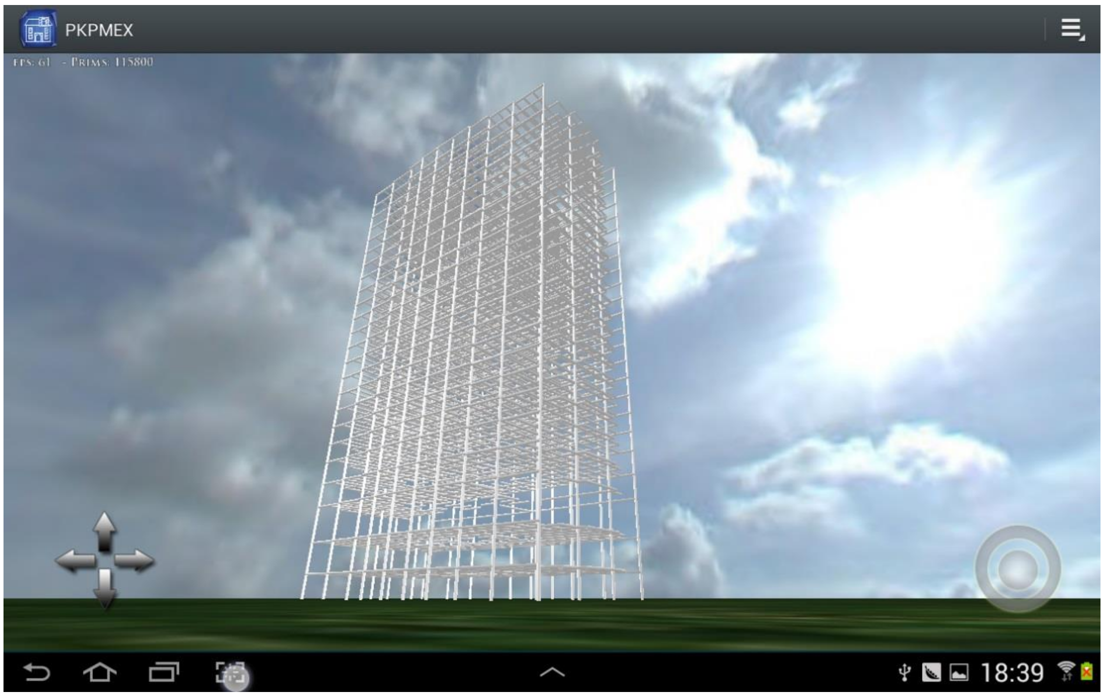

Figure 8: A building model in a mobile device displayed by this prototype.

5. CONCLUSIONS

To solve the problems that the transmission speed of the large-scale 3D-model in a limited bandwidth, as well as platform independence issues to satisfy the practical application requirements, the following technical solutions were obtained based on the authors' studies. The study results can be concluded as follows:

(1) By studying the existed file formats, we find that it is better to use text format to store model data in order to get an easier way for modification and verification, especially in the prototype phase of the project. In the future development, binary file format will be used to reduce the volumn of a building model so that the transmission speed in a limited band width will be increased.

(2) VBO technology resolves the transferring problem between CPU and GPU; greatly use the parallel calculation ability of the GPU. According to another experiment that was done in the Android platform with a 4-core 1.4G Hz CPU, mali-400mp4 graphics card, 30 FPS can be reached when running a model with 150,000 triangles.

(3) If the scene to render is complex, octree collision detection and camera clipping will not be fast enough. The solution is using area segmentation technique and providing a partial model in different precision by the distance (Lod technique). It decreases the complexity of a 3D object representation as it moves away from the viewer so the rendering speed and the visual effects are both satisfied.

(4) Irrlicht engine is a high performance real-time 3D graphics engine written by C++, it can be used in C++ or .Net project. Using Direct3D, OpenGL 1.2 or its own rendering program, Irrlicht can be platform independent.

(5) A building model displayed on a mobile device system developed by the authors indicates that the solutions presented in this paper are effective.

ACKNOWLEDGMENT

The authors would like to thank the project funded by CABR Tech Co., LTD for the supports to this study.

REFERENCES

Hai Na. The development trend of the smartphone. The Software Engineers. 2013.3.15

WANG Jinfeng, YAO Guoqing. Obj Three-dimensional model file format in OpenGL, input and Processing. Computer Knowledge and Technology. 2011.4.5

Huang Xiaofeng 3D drawing method in mobile devices based on OpenGL [j] Industrial Computer control 2013 vol.26

Wu yafeng Android 3D game designing handbook – opengl es 2.0[j] Ren Min You Dian Chu Ban She, isbn 978-7-115-27770-1

Lin Lvping Study on java3D virtual roaming system [d] Xi An Jian Zhu Ke Ji Da Xue, master thesis 2010

Fu Youjia, Yang Kejian Real-time interactive roaming based on the dynamic octree method [j] Journal of Wu Han Li Gong Da Xue 2005