The design of a Software Development Kit for Virtual Reality 3D Audios

Juwon Yun, Dukki Hong,
Dept. of Computer Engineering
Sejong University, Seoul, Korea
{jwyun, dkhong}@rayman.sejong.ac.kr

Cheong Ghil Kim
Dept. of Computer Science
Namseoul University,
Choongnam, Korea
cgkim@nsu.ac.kr

Woo-Chan Park*
Dept. of Computer Engineering
Sejong University, Seoul, Korea
pwchan@sejong.ac.kr

***Abstract* – As the fast growing interest in virtual reality, the demand of technologies associated with them has increased as well. The current virtual reality technology has been developed mainly focusing on visual element; whereas, the sound area has not reflected the surrounding environment properly. There is a limitation to reproduce realistic virtual reality. In order to represent a realistic virtual reality using stereoscopic effect sound technology should be providing a 3D sound. In this paper, we proposed a sound-tracing SDK to simulate the physical characteristics and the environment of the sound in real-time. Because the proposed sound-tracing SDK can be integrated as a plug-in of game engine or provided as a library, developers easily use proposed sound-tracing SDK to reproduce 3D audio and apply the reproduced sound into various mobile environments for virtual reality. We developed demo application using proposed sound-tracing SDK and verified the function of SDK through the demo in mobile device.**

Keywords - 3D Audios;Sound-Tracing;SDK;Virtual Reality

I. INTRODUCTION

Reproducing auditory spatial impression is an important factor for building realistic virtual reality environment. There are several techniques for 3D sound system to provide auditory spatial impression such as Multi-Channel Audio System and Head-Related Transfer Function (HRTF). However, both techniques have limitation of providing auditory spatial impression. Multi-Channel Audio System needs their dedicated speaker system and space for the system to be set. HRTF does not consider surrounding objects and materials of them.

The techniques based on geometric or numeric method have been presented to overcome limitation of generating 3D sound. Ray-tracing based on geometric methods is well known algorithm in 3D Graphics and it can be combined with sound processing technique. This combinational method is called sound-tracing.

Sound-tracing simulates the location of sound source, listener, surrounding objects, and materials to reproduce 3D audio effect. During simulation, the algorithm calculates the characteristics of sound including reflection, transmission, absorption, diffraction. In other words, sound with auditory spatial impression is provided to the listener by reproducing acoustic response in real-time according to environment of virtual reality.

This paper proposes sound-tracing SDK which is provided as a dynamic library or a Plug-in of game engine so that developers can easily apply the sound-tracing SDK to their application.

II. THE PROPOSED SOFTWAFE DEVELOPMENT KIT

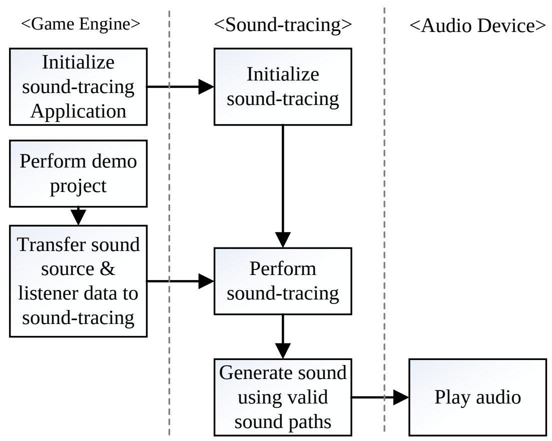

Figure 1. Process for processing a 3D audio sound in a game engine that is integrated by proposed sound-tracing SDK.

The proposed sound-tracing SDK can be used as a plug-in of game engine or a library for other development tool. Sound-tracing SDK links game engine with sound-tracing application and transfers the data between them. The data which is necessary to perform sound-tracing are sound source and listener data. When the data needed for SDK is transferred to sound-tracing application by SDK, sound-tracing is executed.

Figure 1 shows a process for processing a 3D audio sound in a game engine that is integrated by proposed sound-tracing SDK. In the first step, game engine transfers the initializing command to the sound-tracing application. Sound-tracing application initializes the application and loads the sound source, listener data and terrain. In the second step, game engine performs a demo project which is made by developer. And then game engine transmits the information of the object specified as sound source and listener into sound-tracing application in real-time. Sound tracing application performs a sound tracing, by using the data received. In the final step, sound-tracing application transmits the generated 3D audio by simulation in audio device. Audio device outputs a 3D audio which is received from the sound-tracing application.

The tasks for developers among process of sound SDK are as following. The first task is creating a project in game engine. The second task is writing a script to transfer information of the object into sound-tracing application. If the tasks are done, the remaining tasks are performed automatically by sound-tracing SDK.

III. EXPERIMENT RESULTS

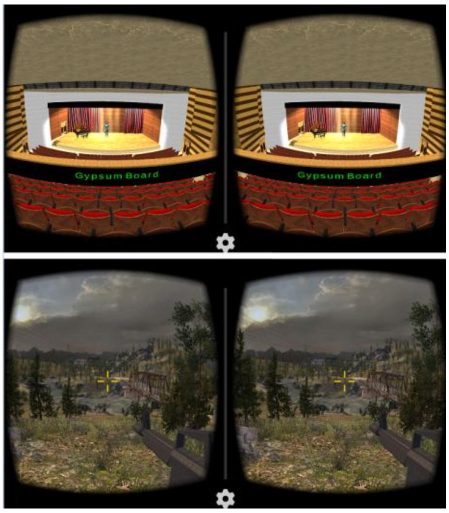

Figure 2. The demo scenes: concert hall and bootcamp

We developed demo for experiment using a sound-tracing SDK. Table 1 shows the result of a demo running on the mobile platform. Figure 2 shows the concert hall and bootcamp demo scenes used in the experiment. Sound-tracing SDK is integrated into the Unity engine in the form of plug-ins. The experiments are progressed in the Android-based Galaxy S6. The factors used in the experiment are CPU utilization, power consumption, and FPS. In the power consumption and CPU usage measurement experiment, we used Trepn profiler which is the profiler application of Qualcomm. FPS was measured by a Unity script. In table 1, ST used a sound-tracing, and non-ST used a game engine 3D audio system.

TABLE1 EXPERIMENTAL RESULT OF THE DEMO USING PROPOSED SDK

	CPU usage	**Power consumption**	**FPS**
Concert hall ST	21.7%	1743.33mW	34
Concert hall Non-ST	19.42%	1475.96mW	43
Bootcamp ST	21.19%	1837.23mW	9
Bootcamp Non-ST	18.23%	1564.03mW	11

According to the experimental results, in the case of using sound-tracing SDK, demo used higher CPU usage and power consumption than case of using Unity 3D audio system. Table 2 shows experimental result when FPS was set to the same value in experiment, power consumption of ST(sound-tracing) is higher than power consumption of non-ST(Unity 3D audio system) about 22~24%. CPU usage of ST is measured higher than CPU usage of non-ST about 3%.

Table2 EXPERIMENTAL RESULT WHEN FPS WAS SET TO THE SAME VALUE

	CPU usage	**Power consumption**	**FPS**
Concert hall ST	21.44%	1648.33mW	30
Concert hall Non-ST	18.79%	1345.96mW	30
Bootcamp ST	21.19%	1926.5mW	9
Bootcamp Non-ST	18.29%	1552.03mW	9

IV. CONCLUSION

We developed demo application using proposed sound-tracing SDK and verified the function of SDK through the demo in mobile device. We were able to reproduce 3D audio with auditory spatial impression as a result. However, a study for reduction of power consumption and CPU usage is needed. Because the proposed SDK can be integrated as a plug-in of game engine or provided as a library, developers easily use the proposed SDK to reproduce 3D audio and apply reproduced sound into various mobile environments for virtual reality.

Acknowledgment

This work was supported by the ICT R&D program of MSIP/IITP. [2016(R7117-16-0193), Development of mobile GPU hardware for photo-realistic real-time virtual reality.

REFERENCES

[1] A. Febretti et al., “Cave2: a hybrid reality environment for immersive simulation and information analysis,” in *Proceedings SPIE Electronic Imaging 2013*, Burlingame, CA , 2013, 864903-864903-12.

[2] M. Vorlander, Auralization: Fundamentals of Acoustics, Modelling, Simulation, Algorithms and Acoustic Virtual Reality, Springer, 2008.

[3] M. Geronazzo et al., “A Head-Related Transfer Function Model for Real-Time Customized 3-D Sound Rendering,” in *Proceedings International Conference on Signal Image Technology and Internet-Based System 2011*, Dijon, France, Nov.–Dec. 2011

[4] V. R. Algazi et al., “The use of head-and-torso models for improved spatial sound synthesis,” in *Proceedings Audio Engineering Society Convention 2002*, Los Angeles, CA, 2002.

[5] N. Raghuvanshi et al., “Efficient and Accurate Sound Propagation Using Adaptive Rectangular Decomposition,” *IEEE Transactions on Visualization and Computer Graphics*, Vol.15, No.5, pp.789-801, Sep 2009.

[6] K. Sunmin et al., “Adaptive Virtual Surround Sound Rendering System for an Arbitrary Listening Position,” *Journal of the Audio Engineering Society*, Vol. 56, Issue 4, pp. 243-254, April 2008.