Figure 0.1: A screen-shot of the test interface design within unity, both in third person 'observer' and first person 'subject' views

An Attempt to Implement a Listening Test to Quantify the Impact of Diffuse Reverberation Level on Localization in Modern VR Applications

Simon Durbridge

January 16, 2017

CONTENTS

LIST OF FIGURES

1 INTRODUCTION

Over the last few years a new wave of Virtual Reality(VR) technology has has become available, often utilising mobile devices as the playback medium. Along with development of the visual component of these VR systems, spatial audio techniques beyond stereo panning and similar abstractions are often being used to create immersive '3 dimensional' sound experiences. In a previous report [?], faculties surrounding the localisation of sound sources within the auditory scene were described. It was suggested that early reflection were a key component used in sound source localisation, and literature suggested that appropriate simulation techniques would be required for accurate sound source localization in VR [?].

1.1 RESEARCH QUESTION

Does the level of artificial diffuse reverberation have an impact on a listeners capacity to localise a sound source in a VR environment, with the use of a simple early reflection simulation method? When utilising a reverberation tool that combines direct reflection simulation and artificial diffuse reverberation, is it possible to counteract the effects of early reflections on localization accuracy by masking the direct reflections with an overly loud diffuse field?

Null Hypothesis: Diffuse reverberation level has no impact on sound source localization error in VR environments.

The aim of this study is to determine if a tool that combines direct reflection and diffuse reverberation simulation, is appropriate from a sound source localisation perspective in VR applications. In this study the Google virtual reality software development kit (VRSDK) [?] for Unity was used for development of the listening test environment. The VRSDK version used was 1.0, and the version of unity used was 5.4.2f2.

1.2 REPORT OVERVIEW

Initially the experiment method will be discussed, and the behaviour of the GVRSDK room reverb algorithm will be explored. Following this, the experiment protocol will be given. Next, cursory experiment data will be analysed, and finally the experimental method will be reviewed and improvements suggested.

2 EXPERIMENTAL METHOD

In order to analyse the tools available for new VR systems, it was considered crucial to develop a listening test that incorporated such a tool with as little modification to the tool as possible. Time constraints were also a factor in realising such a test, as little prior mobile application development experience was directly available. Given these constraints, the Unity game engine system was chosen as a development platform. This was beneficial as Unity has native Android and GVRSDK support [1].

The VR environment the test subject is placed in consists of a single ground plane, with an ancillary menu of simple functions such as Reset function, visual distortion correct and other functions relating to the visual system. The single ground plane appears to stretch out significantly far, to provide limited bias to room size. The reset function allows a user to skip to the next round of the test, if they cannot find the sound source. Test scores for the previous round of the test are also displayed on the plane, so that those administering the test can record the results without having to access the phone.

A $4*4*4meter$ cube sound source with a collider but no direct mesh is randomly moved to a position on a unit semi-sphere of random diameter, centred around the test subject who is held at the same point in space $1.5meters$ above the centre of the plane. The sound source would have a relatively cardioid dispersion characteristic, and would always be oriented to face the test subject. An example of the sound source used can be seen in figure 2.1, with the cardioid pattern, green box collider and viewers reticle (cursor). The reticle is important in VR systems, as it allows users to always understand where their cursor is pointed. To compensate from the visual stimulus of the reticle allowing for easy accurate visual localization, the reticle is allowed to grow relatively small and the box collider is oversized. It was stressed to test subjects that the reticle is a visual guide that would not point directly at the centre of the sound source, and that localization should be based on the sound of the environment with an aim to find the centre of the sound source. Localization error is measured as the angle between the absolute centre of the sound source collider, and the direction in which the test subject is looking.
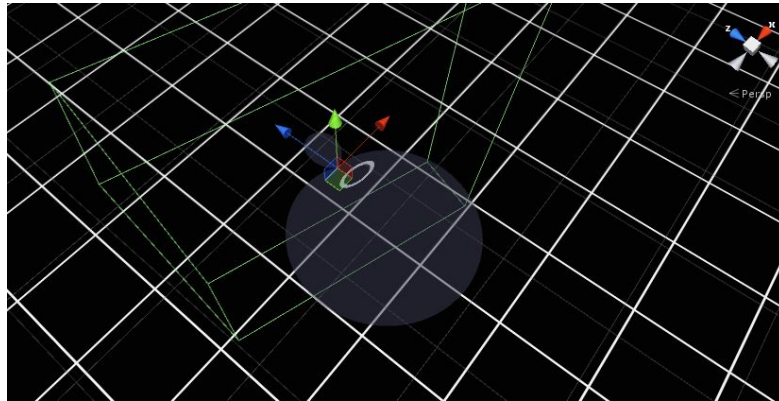

Figure 2.1: A screen-shot of the sound source and its dispersion pattern

As the test progresses the size of the room in the reverberation model is increased, and the amplitude of the diffuse reverb component is varied.

---

[1] Though due to instability, the most recent versions of Unity and GVRSDK were not used in the development of this experiment

2.1 GOOGLE VR SDK REVERB ENGINE

Google have unsurprisingly minimal public documentation discussing how the proprietary GVRSDK GvrAudioRoom component models room reverb. There are however popular methods of calculating low-order direct (early) reflections using methods such as the image-source method, and [CITEME] suggested a reverb modelling method that uses a combination of the image-source method and all-pass reverberation to simulate room characteristics at low computational cost. The user is provided with 11 variable parameters for modelling room reverb:

- Wall Material

- Reflectivity

- Gain

- Brightness

- Time

The first two parameters in the list appear to influence the intensity and length of the early reflection component, as can be seen in figure 2.2. The response shown are of the left receiver, and include the effects of the HRTF filtering used in the GVRSDK. The latter three parameters appear to effect the diffuse component of the reverberation. For the sake of this experiment, only the room size and diffuse reverb gain are altered.
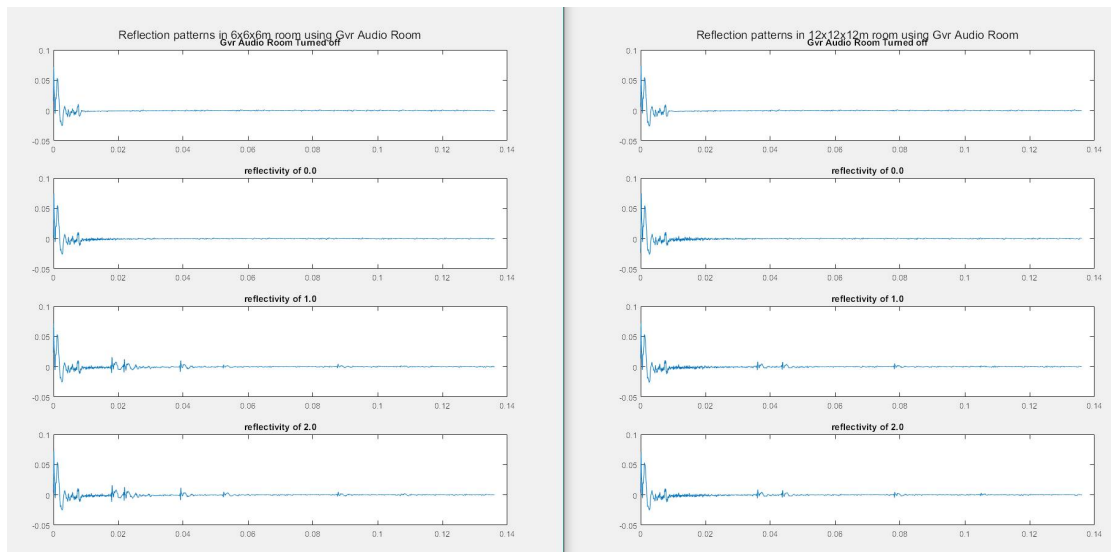

Figure 2.2: Comparison plots of impulse response of 6m and 12m rooms with varying reflectivity and no diffuse reverberation

2.2 Experiment Protocol

Test subjects were asked to evaluate three random source locations in each of three different surrounding room sizes. The room sizes were doubled each time, and the listener position was maintained at $1.5meters$ above the ground plane. The audio sample used was a mono mix of an excerpt from a monologue by climber Chris Schulte, made for the JamCrack Podcast and used with permission of podcast owner Niall Grimes [2]. This audio sample was used as the content was simple and clear, with a reasonable consistency of quiet periods to allow for the decay of the reverb to be heard. The audio room size began at $7meters$ by $7meters$ by $4meters$, and increase to twice as large each time.

3 Results and Evaluation

3.1 Results

3.2 Evaluation

4 Experiment Review

One of the many issues with this test revolved around the need for the reticule as a cursor, thus having to compensate for the reticule by over-sizing the sound source object. The reticule was implemented due to a limited understanding of the triggering mechanics in Unity and the GVRSDK. The reticle must be triggered by hitting the box collider and having the user press the cardboard trigger, in order for the game engine to enter the method for moving the sound source and displaying the results. For an implementation with a relevantly large sample set, a different triggering mechanism should be implemented that is compatible with the 'on trigger' part of the GVRSDK set of functions but is not attached to the reticle directly.

In relation to the reverberation modelling component of the GVRSDK, it would have been preferable to implement an algorithm that included a modifiable early reflection order. This would have allowed for a test of localization error due to reflection order as well as if those reflections could be masked with diffuse reverb.

Later versions of the GVRSDK have some added functionality for the GvrAudioSource and GvrAudioListener objects that are not available in the version used in this test. As these versions of the GVRSDK become more stable, it would be preferable to implement them in a later test.

5 Conclusion

---

[2] http://www.niallgrimes.com/jam-crack-podcast/jcpc-006-the-view-from-dead-horse-point

6 REFERENCES

REFERENCES