UNIVERSITE PARIS-SUD

ÉCOLE DOCTORALE : Ecole Doctorale Informatique Paris-Sud (ED 427)

Laboratoire d'Informatique pour la Mécanique et les Sciences de l'Ingénieur
(LIMSI CNRS) (UPR3251)

DISCIPLINE Informatique

THÈSE DE DOCTORAT

soutenue le 30/09/2013

par

Hui DING

Level of detail for granular audio-graphic rendering: representation, implementation, and user-based evaluation

Directeur de thèse :	Christian JACQUEMIN	Professeur (Université Paris-Sud)
Co-directeur de thèse :	Emmanuelle FRENOUX	Maître de Conférences (Université Paris-Sud)

Composition du jury :

Président du jury :	Anne VILNAT	Professeur (Université Paris-Sud)
Rapporteurs :	Venceslas BIRI	Maître de Conférences HDR (Université Marne la Vallée)
	Stéphane NATKIN	Professeur (CNAM/CEDRIC)
Examinateurs :	Samia BOUCHAFA	Professeur (Université d'Évry-Val-d'Essonne)

Acknowledgements

First of all, I would like to express my sincere gratitude to my supervisor Christian Jacquemin for his valuable guidance and advice throughout this research work. His insight and expertise in computer graphics and multimedia shows me the way to this area. I would like to thank him for his continuous support until the end of my PhD study, as well as his patience. I also thank my co-supervisor Emmanuelle Frénoux for a careful proofread and helpful comments on the thesis.

I am grateful to my research collaborators Roland Cahen and Diemo Schwarz who have given me valuable guidance and shared their knowledge in sound synthesis with me. Many thanks go to my *Topophonie* colleagues and Limsi colleagues with whom I have enjoyed my research life very much.

With a deep sense of gratitude, I would like to thank my parents and Calvin for their endless support and unconditional love. The above list is definitely incomplete. I am indebted to all those who have given me supports and those who have cared me throughout all these years. Thank you all.

Abstract

Real-time simulation of complex audio-visual scenes remains challenging due to the technically independent but perceptually related rendering process in each modality. Because of the potential crossmodal dependency of auditory and visual perception, the optimization of graphics and sound rendering, such as Level of Details (LOD), should be considered in a combined manner but not as separate issues. For instance, in audition and vision, people have perceptual limits on observation quality. Techniques of perceptually driven LOD for graphics have been greatly advanced for decades. However, the concept of LOD is rarely considered in crossmodal evaluation and rendering.

This thesis is concentrated on the crossmodal evaluation of perception on audio-visual LOD rendering by psychophysical methods, based on that one may apply a functional and general method to eventually optimize the rendering. The first part of the thesis is an overview of our research. In this part, we review various LOD approaches and discuss concerned issues, especially from a crossmodal perceptual perspective. We also discuss the main results on the design, rendering and applications of highly detailed interactive audio and graphical scenes of the ANR Topophonie project, in which the thesis took place. A study of psychophysical methods for the evaluation on audio–visual perception is also presented to provide a solid knowledge of experimental design. In the second part, we focus on studying the perception of image artifacts in audio–visual LOD rendering. A series of experiments was designed to investigate how the additional audio modality can impact the visual detection of artifacts produced by impostor-based LOD. The third part of the thesis is focused on the novel extended-X3D that we designed for audio-visual LOD modeling. In the fourth part, we present a design and evaluation of the refined crossmodal LOD system. The evaluation of the audio-visual perception on crossmodal LOD system was achieved through a series of psychophysical experiments.

Our main contribution is that we provide a further understanding of crossmodal

LOD with some new observations, and explore it through perceptual experiments and analysis. The results of our work can eventually be used as the empirical evidences and guideline for a perceptually driven crossmodal LOD system.

Keywords : level of detail for graphics, level of detail for sound, crossmodal perception, audio-visual perception, perceptual experiments, psychophysical methods.

Contents

Introduction **15**

0.1 Motivation 15

0.2 Objective and scope 16

0.3 Approach 17

0.4 Contribution 18

0.5 Thesis organization 19

I Background **21**

1 Overview of LOD management and perceptual issues **23**

1.1 Level of details in computer graphics 23

1.1.1 Simplification forms for 3D graphics 24

1.1.2 Fidelity metrics 30

1.1.3 Perceptually guided LOD 32

1.2 Motivation: Can other perceptual phenomena be modeled? 34

1.2.1 Crossmodal influences on audio perception 35

1.2.2 Crossmodal influences on visual perception 36

1.2.3 Interaction of bimodality in the virtual context 36

1.2.4 Crossmodal perception and multimodal LOD 37

1.2.5 Overall perception 38

1.3 Sound and Level Of Detail 39

1.3.1 Corpus–based Concatenative Sound 39

1.3.2 Dynamic sound level of detail in games 39
1.3.3 Conclusion 40

2 Audio–visual research project *Topophonie* 41
2.1 What is *Topophonie*? 41
2.1.1 Role of doctoral program in the project 43
2.1.2 Organization 43
2.2 The research questions arising from the study of a pioneering work . . 44
2.3 Scene model design from an artistic view 45
2.4 Constitution of modeling method and standard formalism 47
2.5 Building the concept of Sound LOD 49
2.6 System development through collaboration 49
2.7 Applications and products 50
2.8 Conclusion 52

3 Overview of audio–visual evaluation methods and psychophysical studies 55
3.1 Experimental measures of fidelity 56
3.1.1 Perceptually experimental measures 56
3.1.2 Automatic measures 58
3.2 Experimental evaluation and psychophysical methods 59
3.2.1 Procedure of experimental design 60
3.2.2 Psychophysical Experiment–tasks 61

II Evaluation of crossmodal perception on mono LOD system 71

4 Perception of artifacts in audio–visual LOD rendering 73
4.1 Implementation 73
4.1.1 Scenario and graphics rendering 74

4.1.2 Impostor–based LOD algorithm 75
4.2 Experiment design 81
4.2.1 Auditory integration 81
4.2.2 User interface 82
4.2.3 Participants and apparatus 82
4.2.4 Stimuli and procedures 84
4.3 Statistical analysis 85
4.3.1 Evaluation of visual perception of artifact 86
4.3.2 Evaluation of perception of visual artifacts with auditory effect 87
4.4 Conclusion and perspectives 89

III Audio–visual description and modeling 91

5 Spatial audio–graphic modeling for X3D (see Appendix A [Ding et al. 2011]) 93

5.1 Background and motivation 94
5.2 Technical points about X3D 95
5.2.1 Related formalism 95
5.2.2 XML Schema 96
5.3 Audio–graphic modeling principles 96
5.4 Extended X3D representation 99
5.4.1 Representation of sound process 99
5.4.2 Representation of sound source 101
5.4.3 Representation of basic profile functions in 1D 101
5.4.4 Representation of profile functions in 2D and 3D 102
5.5 X3D audio–graphic modeling 104
5.6 Discussion and conclusion 105

IV Evaluation of crossmodal perception on multi–LOD system 107

6 Design and evaluation of audio–visual LOD system 109

6.1 Approach . . . 109

6.1.1 Corpus-based sound synthesis . . . 110

6.1.2 Sound process modeling . . . 112

6.1.3 Impostor/Image–based GLOD . . . 112

6.1.4 Impostor SLOD and integration . . . 116

6.2 Experiment . . . 124

6.2.1 Method of limits . . . 124

6.2.2 Participants and apparatus . . . 125

6.2.3 Stimuli . . . 125

6.2.4 Procedures . . . 126

6.2.5 Analysis and evaluation . . . 128

6.2.6 Conclusion . . . 135

Conclusion 141

Bibliography 144

Appendix 154

A 155

B Extended X3D representation – XML Schema 161

Glossary 171

List of Tables

6.1 Configuration of GLOD and SLOD 123
6.2 Conditions 127
6.3 A sample table for logging the answers of each participant 128

List of Figures

1.1 Mesh simplification operators: edge collapse [Hoppe 1996] 25

1.2 Mesh simplification operator: vertex–removal [Luebke et al. 2002] . . 26

1.3 Five categories for classification of vertices [Schroeder et al. 1992] . . 27

1.4 Shader LOD: textures created by different level of procedural shading [Olano et al. 2003] 29

1.5 Visual contrast sensitivity [Reddy 1997] 34

2.1 Pictures of recording in a acoustic studio at Ircam [Topophonie 2013] 46

2.2 Revolution of a simple curve for 1D/2D/3D profile [Topophonie 2013] 48

2.3 The architecture of IAE and IAEOU [Topophonie 2013] 50

2.4 Topophonie application for iPhone on iTune store 51

2.5 Audio tools integrated in sound maps 52

2.6 The picture of sound sources mapping on two meshes (left), and the picture of foliage clustering for audio–graphic LOD (right) [Topophonie 2013] 53

3.1 Specificity and generality of tasks 62

3.2 Task tree 69

4.1 Illustration of the impostor method applied in tree scene. 76

4.2 Tree of GLOD2 to GLOD5 77

4.3 The left snapshot is the LOD2 and the right snapshot is the LOD3. Impostors are marked by white frames 78

4.4 Illustration of impostor–based tree rendering from LOD3 to LOD4. . 80

4.5 The user interface for the experiment about sound influences on impostor–based LOD 83
4.6 The average scores of LOD5 and noLOD5 are very similar. 87
4.7 The visual contrast on LOD2 and LOD4 88
4.8 The average scores for every LOD with sound and without sound. . . 89

5.1 The proposed element TPSoundProcess inheriting properties from X3DSoundSourceNode.100
5.2 The proposed element TPSound extending X3DSoundNode. 102
5.3 Profile function class hierarchy. 102
5.4 TPGeometryProfileContainer class hierarchy 103
5.5 An example of event chain for an audio–graphic scene 104

6.1 Audio–Graphics Engine and GSLOD Preprocessor 111
6.2 GLODs generation by k–means clustering 114
6.3 The tree scene in three GLODs. 115
6.4 The textures for three GLOD. 116
6.5 IAE developable interface integrated in Max/MSP. 119
6.6 A tree rendered in GSLOD1. 120
6.7 A tree rendered in GSLOD2. 121
6.8 A tree rendered in GSLOD3. 122
6.9 The procedure of experiment for the Approach I 128
6.10 Discrimination for the conditions of identical stimulus. 129
6.11 Discrimination for the conditions of LOD2. 131
6.12 Discrimination for the conditions of LOD3. 132
6.13 Box plot for the 9 conditions 133
6.14 Percentage numbers for every stimuli. 134
6.15 Percentages obtained in all the SLOD conditions 135
6.16 Percentages obtained in all the GLOD conditions 136
6.17 Percentages obtained in all the GSLOD conditions 137

Introduction

0.1 Motivation

During the last few decades, the application of audio–visual rendering of 3D scenes is becoming more and more common in various areas such as commercial PC/video games, medical illustration, and education. In consequence, the technologies of computer–generated percepts have grown tremendously for each sensory modality of audition and vision. Indeed, while the complexity and reality of 3D graphics keep increasingly supported along with the growth of computer hardware capability, the study of level–of–detail has also been greatly expanded to struggle with the trade–off between complexity and performance. Instead of manually selecting LOD, regulating simplification with a metric of fidelity becomes the mostly used LOD selection method in the practice of LOD algorithms, specially for a large database which is often the case in 3D applications. As a result, how to measure fidelity is a significant question that has been most asked. To answer the question, both geometry and surface–attribute error metrics are usually considered in LOD optimization process. However, empirical evidences show that the crucial measure of fidelity is perceptual. In consequence, visual perceptual metrics have been studied for guiding level–of–detail frameworks for the last two decades. The first perceptually–driven LOD system is proposed by Reddy at the end of the 90s [Reddy 1997]. Since then, the challenge of this scope is to understand the foundations of perceptual psychology and apply them into the LOD or even to other computer graphics techniques. Likewise, the technologies of audio rendering have been substantially developed for complex 3D sound, in the aspects of source modeling, propagation simulation, digital signal processing, spatialization, etc. Meanwhile, there are also some interesting trials made to handle massive sound sources in virtual environment by using audio perception [Tsingos et al. 2004]. These rapid growths in audio and visual rendering technologies have more or less involved the use of human percep-

tion in their own research community, respectively. However, very few work has been done in exploring the efficacy of crossmodal perceptual system of audition and vision in an attempt to give a guideline for audio–visual display applications, such as selecting LOD, which can be perceived by user, and modeling audio–visual LOD scenes. To the best of our knowledge, the topic of crossmodal LOD has only been studied by Bonneel et al. [2010] with respect to the graphical LOD of illumination form and contact sound simplification, but such a crossmodal perception study has never been done on image–based and polygon–based simplifications (which are the most common methods used in 3D applications) and environmental sound simplification.

0.2 Objective and scope

From the end of the last century, computer image/graphics researchers have begun to study cross–disciplinary knowledge in order to investigate the crossmodal perception phenomena for justifying a certain degree of audio and/or visual fidelity. From the same time period, LOD researchers have also begun the investigation of principled guideline for selecting LOD by using models of human vision system. The exploitation of perceptual system is not only advanced in computer graphics of virtual environment but also in sound synthesis. In 2004, Tsingos et al. [2004] have first introduced an approach to optimize the sound sources in a large virtual environment by using criteria of audio perception. Recently, researchers have begun to investigate crossmodal LOD through perception. A pilot study [Bonneel et al. 2010] has investigated the efficacy of crossmodal LOD by the perception of material similarity in particular, and based on their psychophysical finding, they derive a perceived quality function of crossmodal LOD so that one can choose the appropriate LOD combination of sound and graphics for the best perceived quality of materials. However, there is still a lack of investigation of overall crossmodal perception in the LOD domain. Furthermore, there are very few references that can guide us in generating, managing and representing audio–visual LOD scene. The presented thesis is dedicated to this growing important area. In an attempt to investigate the fundamental limit of crossmodal detail that an audio–visual system should render, this thesis presents a work which is concentrated on the perception of audio–graphics effects introduced by different simplifications, and the overall perception of a crossmodal LOD scene. To be clear, we have studied the crossmodal perception of the

most common artifacts introduced by image–based LOD rendering system for visual modality. Furthermore, we have also studied the overall crossmodal perception on the crossmodal LOD system. Finally, we have introduced some concepts and standards for representing an audio–visual (LOD) scene.

0.3 Approach

In our work, we employed psychophysical methods which analyze the perceptual system by studying the perceived effects on user experience in varying the properties/factors of stimuli along one or more physical dimensions [Cunningham and Wallraven 2011a]. The things that interest us are the relationship between the unimodal/multimodal LOD stimulus and human sensations that can be affected, and eventually a perceptual guideline for audio–visual applications. In attempt to investigate this relationship, we created various audio–visual scenes with commonly–used LOD generation methods such as image–based LOD and polygon–based LOD, and designed a series of appropriate perception experiments in respecting fundamental psychophysical rules. By analyzing the raw data from experiments, we attempted to understand the capability of uni– and cross–modal LOD through the overall audio–visual perception.

Developing a principled scheme for selecting appropriate LOD is one of the reasons why visual perceptual metrics are applied in LOD for graphics [Luebke et al. 2002]. Indeed, researchers and developers hope to reduce details further while accepting the generated artifacts that may not be perceived by a user due to the limit of visual perception. Simultaneously, researchers also find that sound could influence the perceived visual quality under certain conditions [Storms and Usa 2000]. This makes us think whether the advantage of visual perception on accepting artifacts still exists or whether it is even more obvious when multimodal sensations are involved. Holding this question, we generated our first non–realistic audio–visual scene applied image–based/impostor LOD. As the image/impostor–based LOD inevitably introduced various artifacts, we then designed an experiment to investigate the perception of artifacts under the crossmodal conditions.

For investigating crossmodal LOD in modality of audition and vision, we have introduced the notion of LOD into the sound and invented a rendering engine for generating a crossmodal LOD system. We implemented the impostor–based LOD for both modalities and then performed an experiment based on the *method of limits*

in order to explore the relationship between the overall perception and crossmodal LOD. The impostor–based LOD is the method that we applied for sound, which is the state of the art and the most appropriate one for the corpus–based sound synthesis. On the contrary, there are many more graphics LOD algorithms and their variants have been proposed over the last three decades. We did not try all of them. However, for the purpose of generalization, we implemented another widely–used algorithm which is based on vertex decimation, beyond the impostor–based LOD. A series of experiments based on N–Alternative–Non–Forced–Choice task for discrimination threshold has been accomplished for every trial/scene. Through the series of experiments of audio–visual scene(s), we attempt to explore the capability of crossmodal LOD (i.e., polygon–based graphics LOD and impostor–based sound LOD) by means of the overall perception.

0.4 Contribution

As discussed previously, a lot of research has been done on perceptually driven graphics LOD, as well as the audio–visual crossmodal perception. However, as far as we know, the study of the crossmodal LOD perception has just begun recently and is merely on material perception. To the best of our knowledge, our work is the first to investigate the crossmodal LOD from an overall perceptual perspective and provide a guideline for practitioners who render crossmodal LOD scenes.

- Since no one has ever brought forward a concept or system of sound LOD for audio–visual scene, we not only present and realize this novel notion in an up–to–date sound synthesis application but also combine it with graphics LOD rendering as a pipeline for generalizing audio–visual LOD system through a sound and graphics LOD engine.

- We offer a psychophysics study on multisensory perception of uni– and cross–modal LOD system.

- In consequence, the guide based on the result of experiments through perceptual psychophysics methodology is provided for practitioners who need to generate an audio–visual system especially with uni– or cross–modal LOD technologies.

- We also offer a novel concept for formalizing the modeling of audio–visual (LOD) system, especially in X3D format.

Some of our results have been published in :

- Hui Ding, Diemo Schwarz, Christian Jacquemin, and Roland Cahen. Spatial audio–graphic modeling for x3d. In Proceedings of the Sixteenth Annual International Conference on 3D Web Technology, 2011.
- Hui Ding and Christian Jacquemin. Palliating visual artifacts through audio rendering. In Proceedings of Smart Graphics, pages 179–183, 2011.
- Diemo Schwarz, Roland Cahen, Hui Ding, and Christian Jacquemin. Sound level of detail in interactive audiographic 3D scenes. In Proceedings of the International Computer Music Conference, 2011.

0.5 Thesis organization

The rest of the thesis is organized as three parts comprising six chapters and contains the thesis conclusion, a list of glossary, the bibliography, and an appendix, as follows.

Part I We discuss the background and related work.

Chapter 1 We give an overview of the simplification forms of graphics LOD techniques and discuss the related background about sound LOD notions.

Chapter 2 We present the role of this thesis work in the *Topophonie* project which has focused on the up–to–date techniques in the field of audio–visual rendering.

Chapter 3 We present an overview of uni– and multi–sensory perception and uni– and cross–modal LOD management.

Part II : Chapter 4 We present a development of an impostor–based GLOD scene, and a perceptual experiment used to evaluate the perceptual ability on the artifacts of static scene/image through crossmodal sensations.

Part III : Chapter 5 The concept for modeling an audio–visual system is presented in Chapter 5, and the specification of the extended X3D format as well as a guideline of using the modeling method are provided.

Part IV : Chapter 6 We introduce the audio–visual LOD engine that includes the most common graphics LOD and up–to–date sound LOD techniques. The framework of a crossmodal scene rendering based on impostor–based LOD is presented, and the experiment and psychophysical results are described.

Thesis Conclusion Finally, we summarize the overall findings of this thesis in the conclusion and discuss possible future work.

Part I

Background

Chapter 1

Overview of LOD management and perceptual issues

In the field of computer graphics, level of detail, or LOD for short, has been becoming a focus in the domain ever since James Clark firstly specified the notion and its principles in 1976 [Clark 1976]. Practitioners and researchers in computer graphics have never stopped exploiting and improving LOD techniques, from LOD forms to LOD management, from discrete LOD to perceptually driven LOD, from overall frameworks to concrete algorithms. Graphics seems to be a privileged modality for LOD consideration, since LOD has barely been codified or even mentioned in the field of digital sound. LOD is now considered as a practical technology in computer/video games for reducing the computational cost, since sound also plays a very important role in computer/video games. In this chapter, we firstly give an overview of the major forms of simplification for LOD, and then we talk about the error measures for LOD optimization. Finally, we address the perceptual issues of LOD and introduce the conventional method for modeling a vision system with which one may predict the visual perception. Note that here we not only talk about LOD for graphics, but also bring the idea of sound LOD into the field by investigating the related notions, innovative techniques, and potential applications.

1.1 Level of details in computer graphics

One of the core questions in LOD for today's 3D graphics is how to choose the appropriate LOD at any given moment. This question can be actually answered through fidelity measures of LOD rendering by comparing LOD rendering with full

detail rendering, in which some error measuring approaches are generally used. However, researchers have long recognized that empirically the most decisive measure of fidelity is about perceptual quality, taking in consideration the Human Visual System (HVS) characteristics. As a result, they have started to explore and apply the visual perceptual theory on the LOD techniques [Reddy 1997]. It is known that error measuring approaches would help managing the simplification process by choosing the operation that produces the best output quality of the simplification, and choosing the best choice normally with respect to the error metric(s). Note that the output quality of the simplification can never be the same as the original one due to the principles of LOD, so the errors are unavoidable. As a consequence, the perception of errors fairly is the crucial cue in LOD management. The perceptual quality measuring approaches are meant to tell when to switch between two LODs by judging the perception of errors/differences between LODs according to HVS. If we understand how HVS works with error metrics, we may build a model to simulate the HVS in order to analyze the output of an LOD system and tell which LOD to use and when to switch between two LODs.

For measuring errors caused by simplification operations, several error metrics have been explored, ranging from geometric metrics to attributes metrics. Most of the time, an error metric or a combination of error metrics is chosen according to the LOD forms and the purpose of the LOD system. Since LOD forms obviously determine which fidelity measurement to apply, it is worth discussing the basic simplification forms for 3D graphics in detail before reviewing the error metrics and their measurement methods.

1.1.1 Simplification forms for 3D graphics

Simplification is the method that we use to generate LODs by simplifying the original 3D models to less cost models. The geometric simplification is without doubt one of the most investigated methods in LOD forms. Despite that, the study of other simplification forms is also growing for fulfilling different demands of desired application. In this thesis, we are interested in observers' overall perception on graphics–sound LOD (GSLOD) systems, and we have employed the graphics LOD forms, such as mesh simplification and image-based/impostors method because they are mostly used in the field. Different kinds of graphics LOD forms are reviewed in the following section, including mesh simplification and image-based/impostors.

Geometric mesh/polygon simplification

Practitioners have a variety of choices for polygonal optimization frameworks, thanks to work done from local simplification operators to topology simplification operators, and from algorithm design to error metrics [Luebke et al. 2002]. The simplification operations are meant to lose/sacrifice the fidelity, while the simplified representation is hopefully considered to be the "best rendering" of the original objects. As a result, the error metrics, especially geometric error metrics, are used for measuring the output quality in order to ensure the "best" quality of the rendering [Luebke et al. 2002].

Mesh simplification is generally achieved through local mesh simplification operators, and accompanied by global mesh simplification operators if a higher level of simplification over a much more complex mesh is needed. The local mesh simplification operators remove vertices, edges and faces, with respect to the small faces or the local/partial region of the mesh. On the other hand, the global mesh simplification operators simplify the large region of the mesh based on its topology.

The most traditional local mesh simplification operators are *edge collapse*, first introduced by Hoppe et al. [1993], see Figure 1.1.

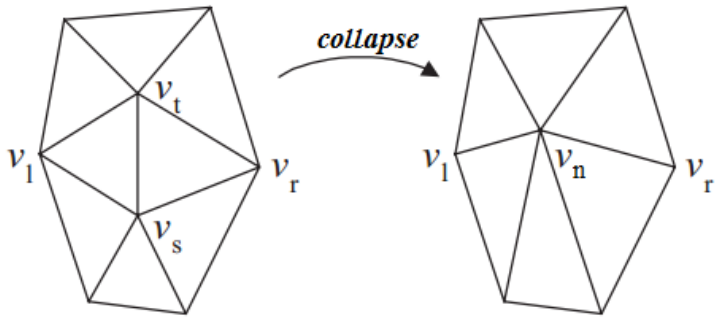


Figure 1.1: Mesh simplification operators: edge collapse [Hoppe 1996]

In the figure, an edge (v_t, v_s) is collapsed by a new vertex v_n. Indeed, there are also some variants of collapse operation. When the replacing vertex is one of the vertices of edge (v_t, v_s), the operator is called *half–edge collapse*. When v_t and v_s are unconnected vertices, the operator is called *vertex–pair collapse* [Schroeder 1997; Popović and Hoppe 1997]. The *triangle collapse* operator is meant to collapse a triangle (v_1, v_2, v_3) into one vertex [Hamann 1994; Gieng et al. 1998]. The *cell*

collapse operator collapses the vertices of a same cell after classifying all vertices into groups [Rossignac and Borrel 1993; Luebke and Erikson 1997].

Schroeder et al. [1992] proposed a *vertex–removal* operator in their *vertex decimation* algorithm. The *vertex–removal* operator is composed of two steps: (i) removing a vertex and its surrounding triangles; and (ii) triangulating the hole. Figure 1.2 illustrates an example of vertex removal operator: when removing a vertex as well as its surrounding triangles, the hole arises; then triangulation is performed to fill the hole.

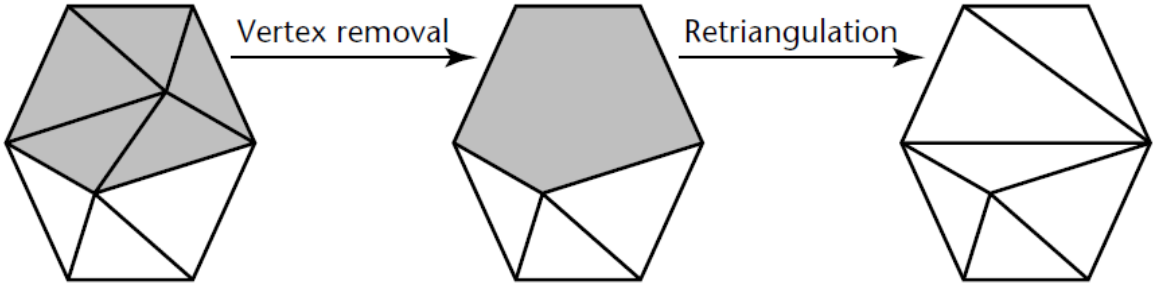


Figure 1.2: Mesh simplification operator: vertex–removal [Luebke et al. 2002]

In practice, *vertex decimation* is an LOD algorithm of good trade–off between efficiency and quality, that basically repeats *vertex–removal* operation for the vertex candidates who meet the specified decimation criteria and stop when it reaches the simplification criterion. Particularly, during every pass of the repeat, it classifies all the vertices of input mesh into five categories as : simple, boundary, interior edge, corner, and complex (see Figure 1.3), and the vertices that do not belong to the complex category are the vertex candidates to be deleted. Then by measuring a geometric error (see Section 1.1.2), the vertex candidates that do not exceed a certain error–threshold are removed along with their surrounding triangles. And finally the triangulation is applied for filling all the holes due to the removals. Due to its process for choosing vertex to remove, the *vertex decimation* is able to preserve the topology of the original mesh. The *vertex–removal* operator may be replaced by *vertex collapse*, and both can be called *incremental decimation* .

The *vertex clustering* is a simple and efficient LOD algorithm. Rossignac and Borrel [1993] were the first to propose the method for simplifying 3D models by clustering vertices. The algorithm is composed of six steps: 1) compute the weight for each vertex based on its perceptual importance; 2) triangulate all polygonal faces; 3) cluster all the vertices in groups on the basis of geometric proximity; 4)

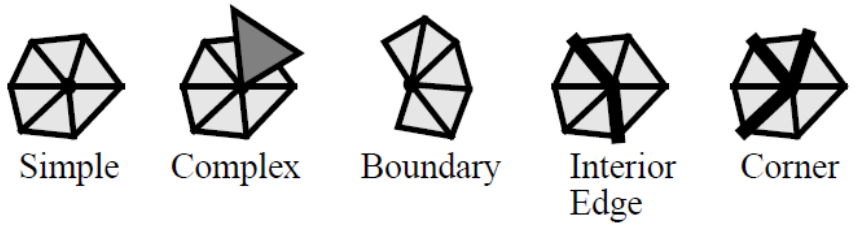


Figure 1.3: Five categories for classification of vertices [Schroeder et al. 1992]

compute for every cluster a representative vertex; 5) remove duplicate triangles, edges, and points; 6) compute normals for new triangles by regenerating vertex–coordinates. Though it is fast, the algorithm cannot preserve the topology, as it runs at vertices without any concern for their connections. On the other hand, it can be run efficiently on messy meshes, such as a disordered foliage.

If an original mesh has a very complicated geometry that contains massive fine details, and if we wish to simplify its topology, it will be necessary to apply a global mesh simplification operator, such as low–pass filtering or morphological operator. The first algorithm to simplify the topology is proposed by He et al. [1996] by applying a three–dimensional low–pass filter to every volumetric subdivision of a given mesh which is divided by a three dimensional grid. Such 3D low–pass filtering, based on digital processing, excludes the fine details such as small holes and thin tunnels, and maintains well the topological features. The morphological operators [Nooruddin and Turk 2003; Williams and Rossignac 2005] simplify objects by applying a dilation operator followed by an erosion operator. Such a closing operation is to remove the high frequency parts such as fine details of objects.

The mesh simplification techniques we have seen so far work well for static meshes but not very well for animation, especially when dealing with deforming objects such as virtual characters. That is why some researchers then have been focusing on such cases. For example, Mohr and Gleicher [2003] present a Deformation Sensitive Decimation method based on the idea of quadric-based simplification algorithm [Garland and Heckbert 1997] for measuring the cost of vertices contraction. Also Mukai and Kuriyama [2007] present an idea of motion LOD, using an LOD control method of motion synthesis with a multi-linear model. Recently, Kavan et al. [2008] introduce a new representation of virtual characters named *polypostors* with 2D polygonal impostors.

Image-based/Impostors method

Apparently, geometric mesh simplification is the most investigated form in LOD research. However, it is absolutely not the only form. *Impostor* (also written as *imposter*) is an extremely practical form of LOD nowadays, which has been first named by Maciel and Shirley [1995] for denoting image–based LOD techniques. Compared to geometric rendering, the rendering of impostors is much faster and it produces a very similar visual output. Its efficiency and good output quality make *Impostors* the very popular LOD method in computer/video games and other visual simulation applications.

The impostor method renders the geometric details through image details, which are simply obtained by mapping textures onto single flat polygon/quadrilateral. When the flat polygons are perpendicularly oriented to the viewer, they are called *billboards*. In practice, some approaches are proposed for efficiently representing complex objects by images, such as z–buffer images [Max 1996], image caching [Schaufler et al. 1996; Shade et al. 1996], layered depth images [Shade et al. 1998], multi–layered impostors [Décoret et al. 1999], and billboard clouds [Décoret et al. 2003]. However, the impostor methods always suffer from various artifacts or overloaded memory storage, or both, basically due to viewpoint change and parallax distortion.

Image–based rendering produces artifacts that distract user's attention and impact visual perception, usually degrading perceived quality. Such artifacts occur when the scene geometry is approximate or when viewpoint differs from the original one, resulting in two main kinds of error effects. One is parallax error, which can be cracks, discontinuity, distortion or other visual deforming differences seen from a different viewpoint. The other one is transitional error, which appears during transition between frames, such as popping, blurring, and ghosting. Note that these two kinds of artifacts are closely related: the main cause of transitional artifacts is the difference in parallax distortions in the images involved in the transition. To reduce artifacts, we may provide more information (depth, multiple layer) to impostors, and update the scene with new impostors when the error becomes perceptually noticeable.

Other forms

The possibility of varying the complexity of shading and illumination to form LODs has long been explored. We may call it material–based LOD, as the il-

lumination models (including various reflection model, lighting model, etc.) and texture mapping are actually used to change the material properties of objects for that type of LOD. For example, by adjusting the number of coefficients of a light–reflection model function such as measured BRDF (Bidirectional Reflectance Distribution Function), different levels of material quality can be generated [Bonneel et al. 2010].

Specially, mipmapping [Williams 1983] can be considered as an alternative example of LOD technique used to manage texture memory. Besides, shader–based simplification in LOD techniques are used to decrease the pixel fill–rate costs by reducing the number of texture accesses in a procedural manner using GPU [Olano et al. 2003] (see Figure 1.4). Specially, they use two principles to reduce the number of texture access: one is to replace a texture access with non-texture-based procedure, called *texture removal*, and the other is to replace one or more texture accesses with one texture, called *texture collapse*.

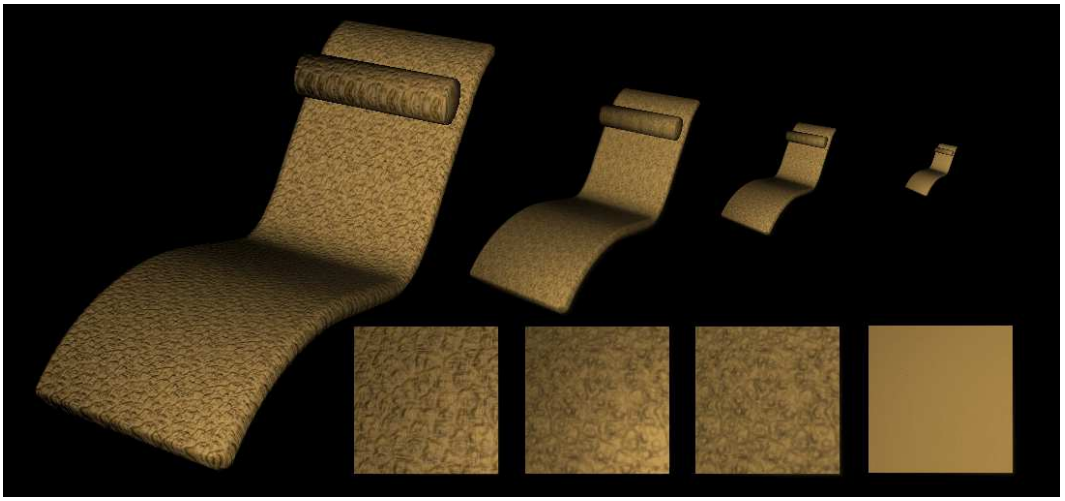


Figure 1.4: Shader LOD: textures created by different level of procedural shading [Olano et al. 2003]

There is another form of simplification that is also classified as an individual technique area, i.e., compression method or also called storage method. For game design and development, the geometry compression [Deering 1995] and texture memory allocation optimization [Dumont et al. 2001] are efficient methods for optimizing the storage of 3D models.

In this thesis, we cannot investigate all existing simplification forms about their efficiency from a crossmodal perceptive. However, we attempt to explore the most

common forms, both in research and commercial area, which are impostor–based method and mesh simplification. The impostor–based LOD method is widely used in game application in order to reduce the computation cost without changing much image quality or to sacrifice the image quality for adapting to different system requirements. Since normally a game is audio–visual, we naturally raise the following question: whether and how a sound, or even some sort of LOD for sound, can influence the capacity of graphics LOD in a perceptual perspective. In LOD field, researchers have long explored the perceptual issues for graphics LOD techniques, especially mesh simplification. However, few work has been done in crossmodal perception consideration. We therefore attempted to investigate crossmodal LOD influences in the overall perception.

1.1.2 Fidelity metrics

Due to the nature of simplification process, all the algorithms that we have seen so far may involve error measuring in order to evaluate the loss of fidelity and to improve the bounds on the error. How to measure fidelity is an essential question to which researchers have provided various ideas and solutions (to be discussed below). The fidelity can be measured by the difference between the simplified objects and the original, and the difference is called *simplification error*.

There are various error metrics and their corresponding measures. A certain LOD system may have a combination of error metrics for minimizing error during the simplification process or in the run–time rendering system. Luebke et al. [2002] gave a very good survey of potential error metrics used in practice.

Geometrical error is the most common error metrics for simplification algorithm, and it is often measured by the screen–space distance between the simplified surface and the original surface. Furthermore, some simplification algorithms also incorporate some measures of attribute errors. Such attribute error metrics can be color, normal, and texture coordinate attributes.

Geometric error Polygon simplification is often evaluated through geometric error measures because of its geometry nature, while impostor method, especially the billboard clouds [Décoret et al. 2003], is also associated with error metrics to give a geometry error threshold on plane selections. The simplest way to measure geometric error is to find the Euclidean distance between two geometry points. However, for the difference between the original and the

simplified object, the distances between vertices and surfaces or surfaces and surfaces are more involved. Consequently, error measurement consists of some types of distance measures, such as vertex–vertex distance [Luebke and Erikson 1997], vertex–plane distance [Ronfard and Rossignac 1996; Garland and Heckbert 1997], vertex–surface distance [Hoppe 1996], and surface–surface distance [Cohen et al. 1996]. Vertex–plane distance efficiently computes the distance between a point and a plane, while vertex–surface distance maps the original vertices to the closest points of simplified surface so that it gives more guaranteed bounds on the error but lower efficiency. The most strongly guaranteed bounds on errors are provided by surface–surface distance which assesses every point of the original surface and the simplified surface in order to find the maximum error to minimize.

Colors In computer graphics, the colors are stored as a three–element array, denoted as (r, g, b). The RGB color spaces are actually three dimensional vector spaces, which provide a simple way to measure color error through Euclidean distance. The difference of two colors, (r_x, g_x, b_x) and (r_y, g_y, b_y), can be denoted by the distance between x and y in Euclidean space:

$$D_{x,y} = \sqrt{(r_x - r_y)^2 + (g_x - g_y)^2 + (b_x - b_y)^2}.$$

However, since the RGB color space is perceptually nonlinear, the distances between two RGB vectors cannot precisely indicate the perceived color difference. CIE L*a*b and L*u*v are two new color spaces that are perceptually more uniform so that the Euclidean distance corresponds better to the perceived difference in color perception by the HVS [Reinhard 2008].

Normals Angular distance (or angular separation) denotes the distance between two normal vectors, in degrees or radians :

$$d = cos^{-1}((n_x, n_y, n_z) \cdot (m_x, m_y, m_z)).$$

Such measurement of the distance of normals can be used for detecting and also avoiding foldover effects due to edge collapse operations [Xia et al. 1997]. It can be also used for selecting good normal vectors at new vertices.

Texture coordinates We store texture coordinates as vectors (u, v) which may be mapped into 2D texture space. So, the measurement of error may be considered as an Euclidean distance between two vectors as Colors error measure (aforementioned) does. The error measurement of texture coordinates may indicate potential fold in texture space caused by some collapse operations, so that a simplification process would prevent them.

As mentioned previously, most algorithms, specially those for mesh/polygon simplification, have adopted measurements of the geometric errors due to the simplification process. Others also consider the effect of simplification process on surface attributes such as colors, normals, and texture coordinates. The error measurement and optimization/minimization operation help reducing differences between degraded object and the original one. However, the LOD process eventually produces a rendering that is different from the original scene due to the essence of degradation operation, no matter how well the error measurement is applied. At last, the final question is how much error/difference a user can perceive? Since any LOD application is ultimately presented to the user visually, the user visual perception of scene should be taken into consideration. Thus, researchers consider that the crucial question of fidelity is not geometric but perceptual: *does the simplification look like the original?* [Luebke et al. 2002]

Note that the error measures are supposed to be an optional optimization of a valid LOD rendering framework, while our work is more concerned about the overall crossmodal perception of 3D scene in order to give a guideline for an LOD method. In fact, researchers have been investigated perceptually driven LOD techniques in visual mono–modality.

1.1.3 Perceptually guided LOD

When to switch between different LODs is actually a perceptual question, which means that, to achieve an optimal LOD, we increase the detail of a degraded representation until we believe that user will visually perceive the change/error of the lower LOD from the original. By heuristics and empirical evidences, the earliest work employed other perceptual criteria than distance and size, such as eccentricity [Funkhouser and Séquin 1993], velocity [Hitchner and Mcgreevy 1993], and depth of field [Ohshima et al. 1996].

The bible of LOD [Luebke et al. 2002] describes the perceptual LOD criteria as follows, quoted:

Eccentricity *An object's LOD is based on its angular distance from the center of the user's gaze, simplifying objects in the user's peripheral vision more aggressively than objects under direct scrutiny.*

Velocity *An object's LOD is based on its velocity across the user's visual field, simplifying objects moving quickly across the user's gaze more aggressively than slow-moving or still objects.*

Depth of field *In stereo or binocular rendering, an object's LOD is related to the distance at which the user's eyes are converged, simplifying more aggressively those objects that are visually blurred because the user's gaze is focused at a different depth.*

However, optimally reducing LOD according to these perceptual criteria should be a prediction process with regard to user perception. Some ideas of modulating LOD based on human visual perception have been proposed. Their essence is in fact to select appropriate LOD by predicting the visual ability to perceive errors, and the thresholds at which a better perceptual model can be appreciated. The human-perceptually noticeable change/error arising from degradation is considered to be made into a visual perception model that should include all the factors that impact HVS [Luebke et al. 2002].

The principles of HVS are based on visual sensitivity that explains the ability of human visual perception. This ability is empirically proved to be represented as a form of curve, which is in turn modeled into a mathematical form called Contrast Sensitivity Function (CSF) (see Figure 1.5). The CSF describes the contrast sensitivity against spatial frequency, and the range of perceptible contrast gratings is shown by the curve. Reddy [1997] first proposed a guideline for modeling perceptual phenomena into CSF model in order to control an LOD system. He introduced several methods to incorporate visual acuity, velocity, eccentricity and other effects into the CSF model which predicts the ability of HVS to perceive visual details. And for implementation, he used image–based metrics to evaluate spatial frequencies on an offline stage. These sampled spatial frequencies were then used to evaluate on an online stage the smallest difference between LODs in frequency form. Finally, with the assessed highest spatial frequency that a user can perceive, the LOD sys-

tem could choose the appropriate rendering parameters so that the spatial changes would not be noticed by users.

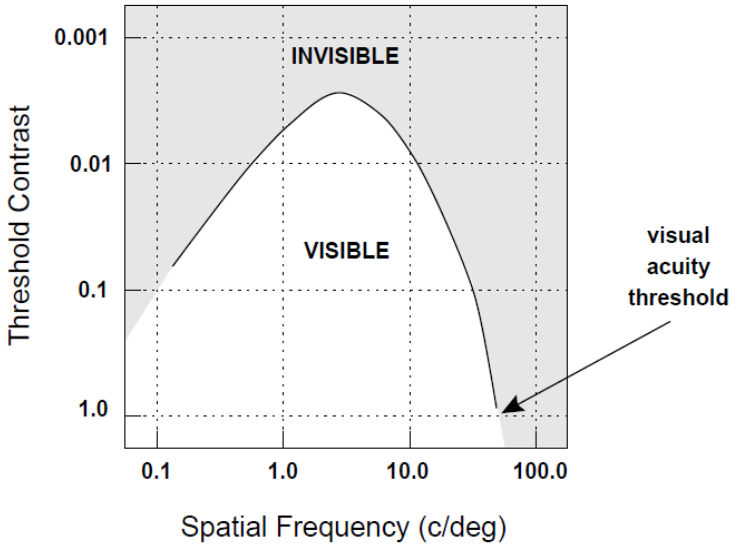


Figure 1.5: Visual contrast sensitivity [Reddy 1997]

Note that vision is a highly complex, highly nonlinear and hardly formulable system. The CSF model is the most applicable tool in graphics rendering for predicting certain phenomena of visual perception with respect to contrast sensitivity. Specifically, CSF is used as a model based on the component spatial frequencies (c/deg) to control the parameters of an LOD system. However, the CSF is concerned only about the vision perception of contrast, and we wonder if we may also model the human auditory system that conceivably affects the overall perception of 3D scene, which means a number of empirical evidences may be needed to provide a new modeling of audio–visual perception or an extended one of CSF.

1.2 Motivation: Can other perceptual phenomena be modeled?

We have seen in the previous section the idea of using perceptual models to predict visual phenomena so as to optimally modulate LOD, and we have also seen the original tool that has been widely applied, CSF model. However, since enormous progresses have been made both in hardware and software, modern 3D applications often support multimodality, especially audio–visual modality. Furthermore, since

audio–visual intersensory phenomena have long been investigated by using experimental approaches [Storms and Usa 2000], it is natural to ask the following question: can we and how do we model a perceptual model for an audio–visual system, especially based on graphics LOD rendering? If the visual perception is assumed to be the basic perception, should we incorporate the auditory phenomena into visual perceptual model? Or the reverse? How to model the phenomena and base model? When auditory modality is integrated with the graphics LOD system, what factors of auditory perception should be taken into account as the causes of impacts of visual perception? If the second modality can also be represented at different details, can it impact the effectiveness of graphics LOD? Can crossmodal perception phenomena be modeled in order to manage LOD by perceptual concern?

We believe that the above questions are highly relevant to computer graphics, especially for modern LOD techniques. In order to answer these questions, it is necessary to think across different disciplines such as philosophy, psychology, neurophysiology, and computer science, to have a great number of empirical evidence, and to derive some algorithmic methods. To the best of our knowledge, these are rarely answered. However, above all, the first thing to do is to investigate the auditory and visual perception of sound and graphics LODs, and we believe that experimental studies are the best approach to do so. Consequently, our work focuses on the crossmodal perception of LODs by analyzing crossmodal LODs system through perceptual experiments and statistical analysis.

1.2.1 Crossmodal influences on audio perception

Evidence shows that what we see impacts our ability of locating a sound. A very famous phenomenon, which can show visual influences on the perception of sound location, is the ventriloquism effect. When a ventriloquist plays his/her trick, the audience perceives that they hear the dummy talking while the dummy is not talking but the performer is. There are some experimental results that verify the ventriloquism effect. Some have found that "visual texture affects the degree of auditory capture of vision, but not the degree of visual capture of audition" [Radeau and Bertelson 1976]. In [Colavita 1974], a series of experiments shows that when presented with combined audio and visual stimuli, in which audio stimuli consist of tones and visual stimuli consist of light flashes, participants think that only flash light occurred. Based on the early work, human visual sense was considered as the

dominant sense when compared with hearing .

1.2.2 Crossmodal influences on visual perception

Recently, a large number of behavioral studies have shown that visual perception can be impacted by other senses. Some have shown that a pulse noise sound can significantly enhance the perceived brightness of an LED [Stein et al. 1996] [Odgaard et al. 2003]. And some have quantified the enhancement of perceived intensity by sound in various contrast detection tasks [Lippert et al. 2007]. Some have shown that sound can help directing attention [Driver and Spence 1998] [Spence and Driver 1996]. In addition to the previous studies of crossmodal influences on static visual perception, there are also some studies of that on visual motion perception [Hidaka et al. 2009] [Valjamae and Soto-Faraco 2008]. More interesting behavioral studies and findings can be found in the research review [Shams and Kim 2010]. From their viewpoint, vision is not considered as the dominant sensory modality that is independent to the other senses. Scientists have recognized that vision illusion can be induced by non-visual information, such as sound.

1.2.3 Interaction of bimodality in the virtual context

In the context of HCI (Human Computer Interactions) or VE (Virtual Environment), researchers are interested in exploring crossmodal perception based on evidences of behavioral studies. The quality of realism in virtual environments is typically considered as a function of visual and audio fidelity mutually exclusive of each other, when researchers explore audio visual VEs [Barfield et al. 1995]. Storms and Zyda [2000] use three perceptual experiments to investigate the crossmodal audiovisual perception phenomenon. The experimental results show two main phenomena: one is that bimodal audio and visual displays, both in high quality, increase the perceived quality of visual displays relative to the evaluation of the visual display alone; the other one is that low quality auditory displays combined with high quality visual displays decrease the perceived quality of the auditory displays relative to the evaluation of the auditory display alone. They therefore suggest that overall perceived quality (sense of realism) of VEs should be a function of crossmodal audio and visual fidelity inclusive of each other. Recently, Bouchara et al. [2013] have proved, by using three experiments, that coherent visual and auditory blurring effects make people less distracted from focusing on distractions targets than either

unimodal blurring effects.

Inspired by these findings, we wonder what sound effects will bring to an LOD-based scene or VE.

1.2.4 Crossmodal perception and multimodal LOD

As we know, a perceptually driven LOD is considered as the best reliable approach to answer the question of "when to switch between different representations for a model". However, we have not found many studies focusing on the effect of sound on LOD capabilities or multi–modal LOD phenomena.

Vision and audition is the most common association of modalities used to enrich our understanding of virtual environment. Recently, Bonneel et al. [2010] have explored the mutual interaction of sound and graphics on the perception of material similarity and in particular with respect to the quality of LOD rendering. They performed a series of experiments to assess the perception of materials, based on graphics and sound LOD rendering.

Graphic LOD (GLOD) stimuli

In order to generate a realistic material, Bonneel and his colleagues adopted BRDF (Bidirectional Reflectance Distribution Function) by projecting BRDF into a set of basis functions which include Spherical Harmonics, Wavelets, etc. By adjusting the number of coefficients of basis function(s), they were able to generate various material quality to produce various LODs.

Sound LOD (SLOD) stimuli

In order to simulate material sound, Bonneel and colleagues considered the impact event which happens only when there is a collision, and used some physical models providing simulation of deformations [O'Brien et al. 2002]. The method uses a set of vibrational modes whose number can be varied in order to generate sufficient SLOD. In their experiments, each stimulus is therefore a sequence of an audio–visual animation about the impact event.

Perceptual experiment

Bonneel and his colleagues made two different object forms (a bunny and a dragon) in two different materials (gold and plastic), and for each one they generated five SLODs and five GLODs. During one trial, a participant visualized two sequences and was asked to give a number that measures the similarity of these two stimuli. They had $2*2*2*5 = 40$ stimuli so that they could evaluate the similarity between various factors. Among their results, the most relevant ones to our work are those concerning the interactions between GLOD and SLOD on perceived material quality. They find that material quality is considered to be higher when SLOD is higher, for the same GLOD. They suggest that, based on these results, the computational cost can be reduced by degrading GLOD and upgrading SLOD, but still maintain the same perceived quality.

1.2.5 Overall perception

To the best of our knowledge, perception of materials is the only topic that has been discussed for the crossmodal LOD investigation. In our work, we attempted to cover more general cases, such as impostor–based LOD, and polygon–based LOD with an up–to–date SLOD which is an original work in our research (see Section 6.1.4.2 and Section 6.1.4).

Following the above review on the artifacts arising from impostor–based LOD and error measuring of polygon–based LOD, we raised four main scientific questions:

i) Can user perception of artifacts due to impostor–based LOD be affected by a simulated, informative sound? How?

ii) Can user perception of artifacts due to impostor–based LOD be affected by a simulated, informative sound at a different level of detail? How?

iii) Can polygon–based LOD be combined with an informative sound with LOD? And does the effectiveness of LOD from one modality influence by the other? How?

iv) If there is a relationship between visual LOD and auditory LOD, can it be applied to control rendering parameters for LOD selection?

For exploring these questions, it is worth taking a look at the sound LOD background.

1.3 Sound and Level Of Detail

Digital sound is a complete and well developed domain in computer science. However, the term *Level Of Detail* rarely arises in this domain. LOD has been a practical and proven technology in computer graphics for balancing the complexity and performance, while it is not a common issue in audio processing and rendering. To our knowledge, the discipline of LOD for sound (SLOD) is only practised in some audio–visual interactive applications such as video games for which the audio modality is always integrated with 3D graphics to offer a more striking user experience.

In collaboration with an expert of digital sound, Dr. Diemo Schwartz of IRCAM, we proposed an original conception of SLOD and then realized the SLOD into audio–graphic rendering. This novel technique has been applied in most of the audio–graphic applications of project Topophonie (see Chapter 2). Our SLOD system is based on the Corpus–based Concatenative Sound synthesis, which will be introduced in the next section.

1.3.1 Corpus–based Concatenative Sound

Schwarz et al. [2006] have proposed a novel sound synthesis method, called Corpus–based concatenative synthesis (CBCS) (see also in Section 6.1.1). This method slices the sound sample(s) into sound snippets, and recomposes them in a manner of parametric selection to make a desired sound [Schwarz et al. 2006; Schwarz 2007]. The audio rendering applied in our work is based on such a novel technique, because it is well suited to render granular audio and graphical scenes such as trees or rain.

1.3.2 Dynamic sound level of detail in games

As we know, the only area where SLOD has been mentioned is the game industry. Game designers have noticed that sound sources can be less used/selected from an original number of sound sources when the auditory objects are moving further from

observer instead of filtering attenuated recorded sources. For example, a gun shot near the observer should be realized through a wide variety of contact sounds of metals. However, when it is in the distance, it should be much vaguer. In this case, a repeated sound snippet is sufficient to generate a shot simulation.

In this thesis, we attempt to bring forward the concept of sound level of detail (SLOD). And we also apply it for audio rendering in audio–graphic scenes by using corpus–based sound (see Section 1.3.1). The detail of SLOD and its application are introduced in Chapter 6.

Our research is an essential part of the french ANR project *Topophonie* with which we first propose the conception of sound impostor so as to realize the SLOD technique based on Corpus–based Concatenative sound.

In order to generate sound impostors [Schwarz et al. 2011] at a given LOD, it is necessary to record a limited time of a mix of audio sources of higher LOD (see the details in Section 6.1.4). Such sound impostors are suitable for generating tree SLODs since audio sources can be well associated with tree branches (see Chapter 6).

1.3.3 Conclusion

So far we have seen briefly the related work that has been done on the perceptual issues of GLODs and in the crossmodal perception of an audio–visual system. Furthermore, we have raised our research questions in the crossmodal perception of GLOD and GLOD&SLOD. We have argued that it is necessary to evaluate the perception of GLODs and GLOD&SLOD through perceptual experiment methods. For doing so, serious studies on psychophysical methods and experimental design are needed for our work. In the next chapter, we present an overview of such cross–disciplinary knowledges and approaches that concern essentially our work in perceptual evaluation of LODs.

Chapter 2

Audio–visual research project *Topophonie*

This PhD thesis is an important part of the ANR project *Topophonie* consisting of research partners from different disciplines. In this chapter, we will present how the PhD program works in the project *Topophonie* and we will also discuss the main products developed in *Topophonie*.

2.1 What is *Topophonie*?

Funded by ANR (French National Research Agency)[1], *Topophonie*[2] is a research project gathering experts from various disciplines to work on diverse aspects of audio–visual scene rendering for the purposes of academic research, art designing, and engineering.

The project *Topophonie* aims to explore new concepts and methods for designing and improving interactive audio–graphic scenes. The work is not only done for academic research, but also applied to innovative and practical products. This creative project groups researchers, artists, and engineers from different laboratories and R&D companies:

ENSCI – Les Ateliers Researchers from ENSCI (Ecole Nationale Supérieure de Création Industrielle) are experts on digital arts and design. They specialize in analysing behaviors or processes in real life and then describing and modeling

[1] http://www.agence-nationale-recherche.fr/
[2] http://www.topophonie.fr/

them in a manner of digital abstract–art, specially for sound.

LIMSI Researchers from LIMSI bring ideas and advices on computer graphics for *Topophonie*. *Topophonie* supports the PhD research project which is enrolled in the laboratory LIMSI (Laboratoire d'Informatique pour la Mécanique et les Sciences de l'Ingénieur) and the University Paris–Sud. The doctoral program attempts to explore the crossmodal LOD in the audio–graphic context by collaborating with other partners of *Topophonie*. The research results of the PhD are all documented in this doctoral thesis and this chapter offers an overview of the collaborative context in which it took place.

IRCAM Researchers and engineers from IRCAM (Institute for Research and Coordination of Acoustic Music) specialize in sound synthesis and design.

ORBE ORBE is a software design and development firm, which specializes in mobile application.

NAVIDIS ORBE is a software design and development firm, which specializes in digital maps for local communities and services.

USER STUDIO Practicers and engineers from USER STUDIO, conceived of original ideas, are interested in developing innovative products and services.

The *Topophonie* partners have developed sub–projects which require collaborations by exchanging ideas, co–designing scenes, and co–developing tools. Together, Topophonie partners have explored new ways of producing synchronized sound and graphics in interactive audio–graphic context, through scientific research, conception, software engineering, and artistic modeling.

Scientific research: studying the state of the art, formulating research problems, defining concepts, developing system architecture, improving standard format, and evaluating systems from a perceptual perspective.

Conceptual work: designing and modeling concrete audio-graphic scenes inspired by artistic creation.

Software engineering: implementing products for innovative usages or services.

Creation: creating of audio–graphics models, demonstrators and artistic works.

The PhD program has mainly focused on the scientific research and it has also involved all the other categories of work through collaborations with the partners.

2.1.1 Role of doctoral program in the project

This PhD program is supported and targeted for *Topophonie*, and its purposes in the multi–disciplinary project are as follows:

- providing scientific study and suggestion of the state of the art,
- co–creating conceptual ideas and methods for modeling,
- co–drafting conceptual convention and language schema,
- developing graphics LOD system,
- co–developing GSLOD system,
- designing experimentations based on psychophysical methods for evaluating crossmodal LOD in audio–graphic context,
- performing data analysis, answering scientific questions, and providing feasible guidelines for interactive audio–graphic applications based on perception evaluation.

The PhD program is oriented towards *Topophonie*'s purposes and has the autonomy of research interests. Meanwhile, some primary decisions of *Topophonie* are made for the interests of PhD program or based on the orientation of PhD program.

2.1.2 Organization

In this chapter, we present how our PhD program serves for the project *Topophonie*, and how *Topophonie* adjusts research orientation based on PhD program's interest and focus. The details of the work and results of *Topophonie* can be found in the project report [Topophonie 2013].

The remaining of the chapter is organized as follows. Section 2.2 introduces how we find our research question from a recent pioneering work based on perception of materials with respect to an audio–graphic LOD rendering. According to the research focus of perception of crossmodal environment, Section 2.3 shows how artistic

designers of *Topophonie* present their ideas about audio–graphic events in real world to researchers and engineers. Inspired by modeling design work, we have noticed the requirement of a modeling method for audio–graphic scene and a standard formalism for exchanging 3D data between different platforms. So in Section 2.4 we present a modeling method which takes into consideration sound processing and an extended scene representation format. Section 2.5 then shows how we collaborated with a partner in digital sound processing to create the new concepts of Sound LOD. In Section 2.6, we explain the collaborative work for our GSLOD system development based on the innovative concepts that we have created. Finally, we introduce in Section 2.7 three creative applications developed in *Topophonie* based on our newly created concepts.

2.2 The research questions arising from the study of a pioneering work

In the beginning of the project *Topophonie*, we have studied the pioneering work of Bonneel et al. [2010]. We were interested in how they investigated the bimodal perception of materials properties, and we wondered if that investigation could be extended in a more general context, which means the investigation of crossmodal LOD by evaluating an overall perception of an audio–graphic scene rendering. From the viewpoint of a doctoral research, we have shared our knowledge of related work with partners of *Topophonie* and arisen some research questions inspired by [Bonneel et al. 2010]. Therefore, *Topophonie* had decided, from the beginning, that its research focus would cover the innovative topic of crossmodal LOD.

Consequently, the PhD program, at first, has mainly worked on the top of the state of the art in three different disciplines:

- Level of detail for 3D graphics,
- Perceptually driven LOD for graphic and audio–graphic rendering,
- Experimental design and psychophysical methods.

We have studied LOD for 3D graphics, since we attempt to investigate the overall perception of general LOD rendering which is not the case in the work of Bonneel et al. [2010]. We have studied the work on perceptually driven LOD for graphic

rendering, from which we have found that conventional work and results on this topic are generally all based on the CSF modeling of HVS, and this approach is rarely used for audio–graphic rendering. Therefore, we have argued that empirical evidences are needed for guiding the audio–graphic rendering through perception–based control of crossmodal LOD. This is why we have also studied the experimental design approaches based on psychophysical methods.

The PhD work program has been presented and motivated to the *Topophonie* partners, and collaborations have been proposed to the partners (scene designers, sound researchers, developers, etc). Scene designers then have illustrated interesting behaviors and situations of real world in an artistic way (see Section 2.3). According to GLOD, sound researchers brought an idea about how to constitute SLODs based on their sound synthesis method (see Section 2.5). In the collaboration with scene designers and sound researchers, we have created the audio–graphic modeling method based on the concept of *sound process* (see Section 2.4), and then developed the audio–graphic LOD systems (see Section 2.6). The tree GSLOD system, which is based on impostor–based GSLOD, is described in Section 6.1.

2.3 Scene model design from an artistic view

Topophonie designers and designer students have worked on sub–projects to create different audio–graphics models. Certain sub–projects have focused on how to represent the foliage scene in audio–visual modality, which have inspired us to develop the tree GSLOD system (see Chapter 6). Others have worked on granular audio–graphic flows, sound maps, and so on.

Audio–graphic focusing and blurring

The idea is to represent the foliage by using image and audio blurring when it is out of focus or far away from the observer. Depending on the position of the observer, the foliage is represented by blurred sources. The photos of real leaves are edited for blurring effects, while leaves sound recording are blurred by adding white noise. Such an idea is considered as a GSLOD representation. The GLOD is a kind of image/impostor–based GLOD which replaces a finer photo with a blurred impostor. And SLOD can be considered as a filter–based SLOD. This combination of GLOD and SLOD is likely low–cost and easy to apply. However, if blurring sources in real time, we should make sure that the

cost of blurring sources is less than that of the original.

Crossing foliage

In this sub–project, designers have tried to analyze the audio–graphic characteristics of foliage through video models. They have studied two cases: (i) the wind and (ii) a person, going through the foliage. These two cases were translated into two different sound activation profiles in graphics: a point is used to simulate a human hand, while a line is used to simulate a wind path. This simulation of activation profiles was rendered in Unity3D and used sound samples that were recorded in real world in the Ircam studio to ensure high quality (Figure 2.1). According to this work, they have conceived the idea of *sound process* and *activation profile*.

Figure 2.1: Pictures of recording in a acoustic studio at Ircam [Topophonie 2013]

Modeling rain

This sub–project focuses on rain particles sound generation with respect to the collision with different surface materials: street, pavement, bus shelters, and mud puddle. In Unity3D, a "hyper–localized" rain is rendered in the real time scene by using a *particle–system*. The sound of rain is triggered at each collision of a raindrop with a contact surface. In this sub–project, designers concluded that rather than producing thousands of raindrop collisions in the overall scene, it is sufficient to generate only an area that is visible to and around observer. In this model, they also simulated three LODs for sound. The LOD1 is "hyper–localized" rain with singular impact calculations for sound. The LOD2 is statistical calculation of the number of impacts, while the LOD3 is ambient sound of distant rain. The difference with the tree is worth to outline, in that these three LODs are produced simultaneously while there is only 1 LOD at a time for the tree. This is due to the fact that rain occurs

in the whole scene while a tree is localized. However, for modeling a forest, a technique similar to that for modeling rain could be used.

Based on these sub–projects, we have extracted the modeling ideas for rendering our concrete GSLOD system. We have chosen to apply the wind profile activation for a tree scene as the scenario, which is sufficient to investigate the capability of GSLOD from a perceptual perspective.

2.4 Constitution of modeling method and standard formalism

The artistic design of scene models have provided ideas and advices for developing the concrete audio–graphic system simulating the real world behaviors. We have noticed that the modeling techniques for auditory scenes and graphical scenes have been largely developed in their community, respectively. However, to the best of our knowledge, there is no guideline for modeling a general audio–visual environment, especially audio–visual LOD system. Thus, we believe that generating a basic concept for audio–graphic modeling is necessary. Based on the investigation of scene models design, an idea of *activation–profile*–based *sound process* has emerged.

Topophonie attempts to develop schemes to model audio–visual events of real world in virtual environment. Roland Cahen, a sound designer and researcher at ENSCI, is the head of *Topophonie*. He has proposed an architecture to model sound sources, activators, and observers together for representing audio–visual events. The observer is surrounded by audio–graphic objects which can be modeled as sound sources. Sounds are activated through the observer behaviors or other active elements, and such behaviors or active events are named activators in our work. Each of the three elements (sound sources, activators, observers) is shaped according to a profile that defines the activation of sources by the observer or active events. Such an activation profile is designed as a geometry form in 1D, 2D or 3D, so that its interaction with some of the preceding three elements generates audio–visual events. How they are activated is determined by the parameters of the profile and physical contacts of geometry objects, which will be discussed in detail in Section 5.3. Roland Cahen has proposed that a simple curve can be itself a 1D profile and be derived to 2D and 3D profiles (see Figure 2.2).

Based on the above idea, in collaboration with one of our partners, Diemo

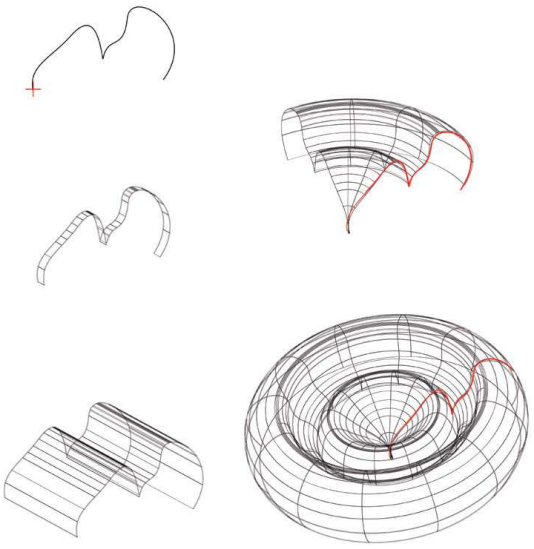


Figure 2.2: Revolution of a simple curve for 1D/2D/3D profile [Topophonie 2013]

Schwarz, we have then developed diverse 1D profile functions and the transformation approaches (see Section 5.3 and Appendix A) for a concrete specification of audio–graphic modeling method based on *sound process*.

Meanwhile, during the collaborations, we have noticed that the lack of standard format caused many difficulties in exchanging 3D data between different platforms. So we have proposed two modern file formats based on XML for storing/representing 3D content: X3D and Collada. By comparing the functionalities of the two formats, we have chosen X3D due to its extensibility which allows us to realize our concepts about *sound process* into a standard format (see Chapter 5).

We have also compared two mostly used XML schema languages, XSD and Document Type Definition (DTD). DTD is more relative and native to the XML specification. However, it has a limited capability compared to XSD. Besides, in an XML–based format, XSD is more suited for extending the XML schema for a complicated usage, such as a complicated audio–graphic event representation. In Chapter 5, we have presented the XML schema in XSD; moreover, we have also shown the relative specification which uses in fact the same style format as DTD's.

Note that for our development of tree scene, we have used Xerces C++ for importing and exporting a valid X3D file.

2.5 Building the concept of Sound LOD

We proposed our idea of GSLOD and discussed with partners of *Topophonie*, and we realized that SLOD is rarely mentioned in the field of sound synthesis or sound design. In collaboration with Diemo Schwarz, we created the concrete concept of SLOD in detail. This contribution is presented in ICMC2011 [Schwarz et al. 2011]. We firstly brought the concept of SLOD by designing three basic LODs for a general case of sound representation, as well as a proposition of the transition approach between different SLODs. Based on the innovative work of SLOD, we have developed the GSLOD system, as well as the evaluation work on this system (see Chapter 6). Note that the corpus–based sound synthesis, developed by Diemo Schwarz, is used in almost all sub–projects of *Topophonie*. This sound synthesis is also suited for implementing the SLODs in a concrete audio–graphic rendering for our experimental development (see Chapter 6).

2.6 System development through collaboration

We developed our first tree GSLOD system based on *sound process* in collaboration with Diemo Schwarz of Ircam. By showing a demonstration of Woody3D tree scene, we have found that the wind function can be easily determined at the same time as an activation profile. So, by applying Woody3D API tool and IAE engine, we designed an audio–graphic engine for developing the audio–graphic system based on *sound process* (see Chapter 6). The Audio Engine and Graphic Engine are developed in two computers through different development tools, VC++ and MaxMSP, respectively. Both engines communicate through OSC messaging. Certainly, it is feasible to move Graphic Engine to the MacOS system of Audio Engine.

We then investigated the capability of crossmodal LOD through our GSLOD system. And we believe that the crucial method for this investigation should be better in a perceptual manner. So, we studied psychophysical methods to evaluate GSLOD through perceptual experiments. For experiencing different techniques, we have also rendered the tree scenario into Unity3D by using different SLOD and GLOD generation techniques.

2.7 Applications and products

We have described so far the collaborations that have been done for research purposes. In this section, we will list some of the creative applications and products that have been produced in the framework of *Topophonie*.

IAE and IAEOU

We have mentioned the IAE engine previously. It is an audio engine for corpus–based synthesis developed in MaxMSP by Diemo Schwarz at Ircam. It has been integrated into a game engine, Unity3D, to generate interactive virtual sound sources based on recording samples. This Unity3D plugin based on IAE is named IAEOU (IAE Object for Unity). A sound file is segmented into a sequence of sound units in IAE. Such a segmentation and the description of sound properties compose the annotation procedure in IAE. Based on the annotated audio materials, IAE performs granular and concatenative synthesis. This IAE engine was adapted for our concrete tree scene for generating SLOD sound (see Section 6.1.4.1). The IAEOU is a modified version for the usage in Unity3D, by abstracting IAE descriptor and audio parameters with presets and profiles. The architecture of IAE and IAEOU is presented in Figure 2.3.

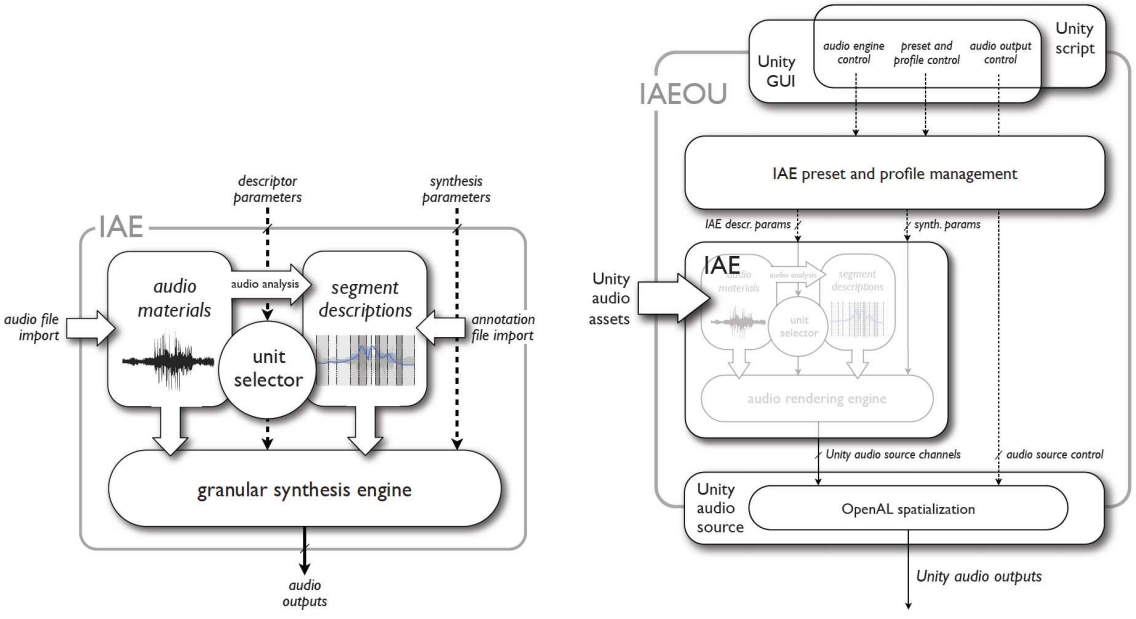


Figure 2.3: The architecture of IAE and IAEOU [Topophonie 2013]

Audio augmented navigation in mobile

The IAE is also used in the mobile application developed by ORBE in *Topophonie* to generate water sounds. This mobile application (see Figure 2.4) allows people to perceive diverse water sounds of a real urban space when wandering in the area. This innovative product offers a new polyphonic and immersive experience in an augmented reality sound environment.

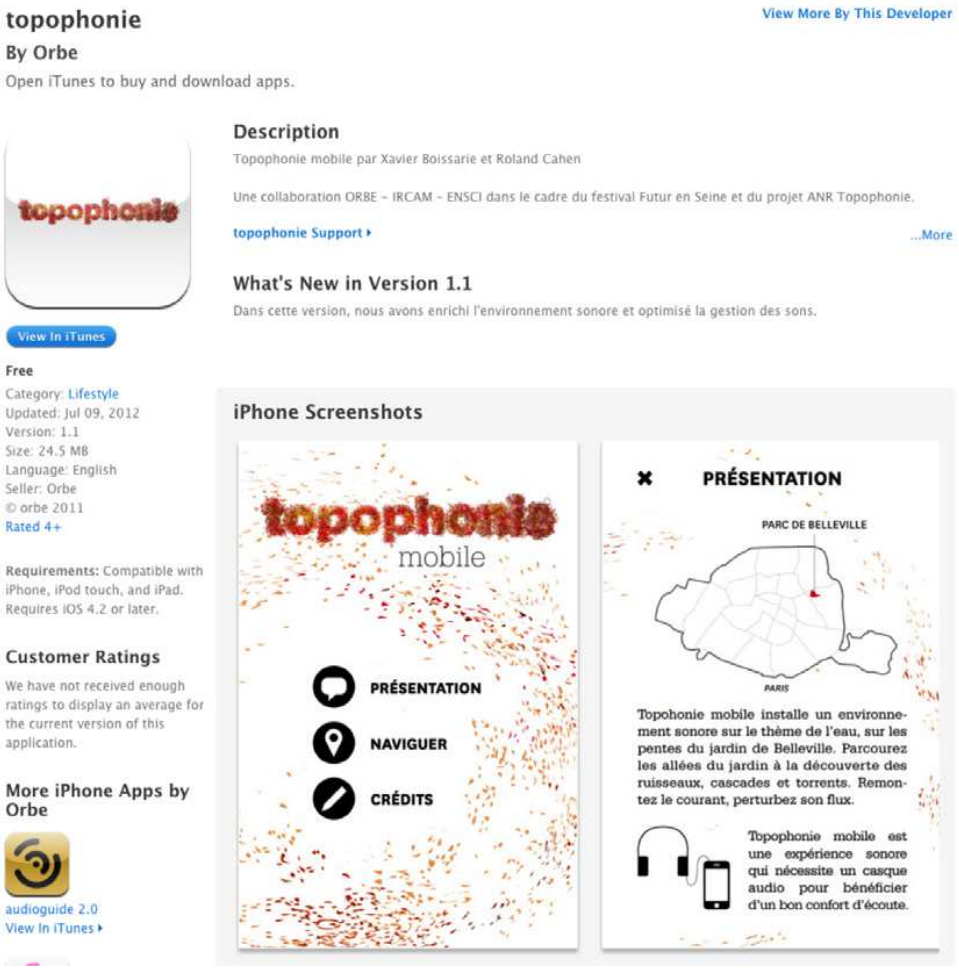


Figure 2.4: Topophonie application for iPhone on iTune store

Sound maps

Navidis has worked on an application for sound maps in order to provide users an acoustic dimension while exploring in territory map. The application is developed based on a community mapping platform, called Navidium, which provides a series of media tools for adding sound and video content to maps

and manage them in free mode (see Figure 2.5). This offers a new experience of consulting territory maps, which shows a wide scope of interests of *Topophonie*.

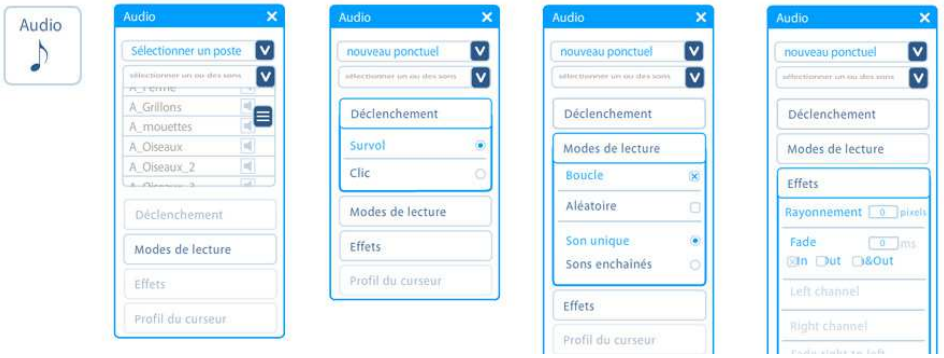


Figure 2.5: Audio tools integrated in sound maps

Topophonie Unity software library

Most of the applications of *Topophonie* are developed in Unity3D. A Unity3d supported software coding library is necessary for our audio–graphic rendering. The Topophonie Unity library, developed by Jonathan Tanant, mainly features the mapping of sound sources and management of GSLOD (see Figure 2.6). By building diverse modules supporting complex graphics output, Jonathan Tanant has created the software library for fulfilling Topophonie's engineering task.

Jonathan Tanant has developed a GSLOD system in Unity3d based on decimation simplification. The GLOD of tree scene is rendered by applying an algorithm of vertex collapse, and SLOD is rendered by a dynamic sources clustering method that computes in real time the number of sources based a sound source mapping and a listener property. Both GLOD and SLOD methods are built in the Topophonie Unity Library. In collaboration with Jonathan Tanant, we designed and developed the experience UI for evaluating the GSLOD through perceptual perspective that is not described in this thesis.

2.8 Conclusion

The PhD program is a part of the *Topophonie* project. We have formulated research questions and conducted work either autonomously or in collaboration with partners. Most results of the project have been produced through cross–disciplinary

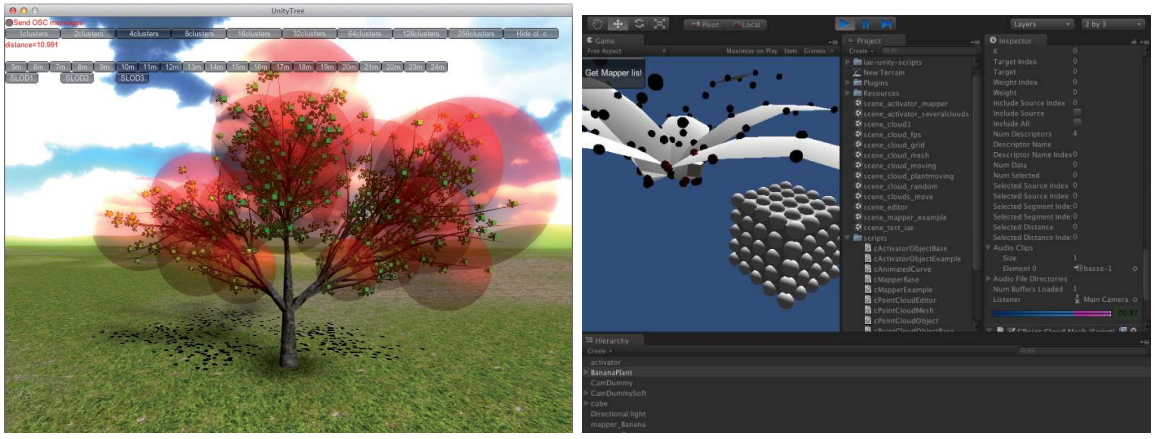


Figure 2.6: The picture of sound sources mapping on two meshes (left), and the picture of foliage clustering for audio–graphic LOD (right) [Topophonie 2013]

collaborations between the partners, including our contributions under the PhD work presented in this thesis.

Chapter 3

Overview of audio–visual evaluation methods and psychophysical studies

We have argued in Chapter 1 that the crossmodal perception of an audio–visual virtual environment is how a human perceptual system functions by processing audio–visual stimuli through multisensory modalities. Some researchers have found some crossmodal perception phenomena in audio–visual displays that eventually impact the overall perception (see Section 1.2). We therefore wonder whether there are certain confirmed mutual–interactions between audio perception and visual perception, which impact the overall perception regularly. We believe that if there is a solid correlation between crossmodal perception and multimodal LOD, one may use it to guide the LOD algorithm in an audiovisual system. Furthermore, if the correlation can be formulated and modeled, one may modulate an LOD algorithm through multi-modality. To answer these research questions about the correlation between perception and crossmodal LOD, a fidelity assessment of the overall perceived quality of audio–visual environment is needed. As aforementioned (see Section 1.1.3), the most appropriate measures of visual fidelity should be based on visual perception. We argue that it is true for all modes of sensory system and we attempt to investigate the measuring methods for fidelity in a perceptual perspective.

A recent work of Bonneel et al. [2010] has focused on the crossmodal perception of material rendering quality using various LODs for sound and graphics, which we have explored in Chapter 1. The essential part of our research is inspired by their work, and we would like to explore the crossmodal perception of LOD–based rendering in a more common and general case: practical measure approaches and

common LOD techniques with various integration of audio and visual modalities. We have reviewed the most common GLOD techniques in Section 1.1, and now we explore the methods of fidelity measures which, we believe, are adaptable for all modalities, especially audio and visual modalities.

3.1 Experimental measures of fidelity

Since in a virtual environment the observer is human, the best way to measure the fidelity is, as a matter of fact, perceptual assessment. Therefore, it is necessary to measure fidelity by investigating human perception of audio–visual stimuli.

The reason to study experimental measures of fidelity is that once having developed a fidelity predictor based on results arising from a well designed experiment and the data analysis, one may use the predictor to automatically conduct LOD in real time. In this section, we will introduce experimental measures of fidelity, which can be used for perceptual experiment design and data analysis.

3.1.1 Perceptually experimental measures

The experimental measures of visual fidelity are the methods for experimentally investigating perceptual behavior, and the result is hopefully to be inferred as an accepted predictor function. A detailed review of traditional experimental methods in psychophysics can be found in [Elmes et al. 2011; Kantowitz et al. 2009]. These psychophysical methods used in experimental design are also widely used in computer graphics. Based on the study of practical cases in computer graphics, Luebke et al. [2002] have summarized several basic experimental measures of visual fidelity, including searching, timing, threshold, ratings, etc. In our consideration, they can be used to evaluate auditory sensing, or even other senses.

- Searching targets: By asking people to search for a target (a specific sound, a specific object, or an event, etc.) in stimuli, one can compute the search performance that actually measures the fidelity of concerned stimuli. To measure fidelity by searching task, there can be two criteria in the evaluation: *time* and *accuracy*. We count the elapsed time of searching period, which is the time from the beginning moment of searching to the end of searching. Besides, we record the accuracy of locating a present target or claiming an missing target.

We can use individual criterion or both the *time* and the *accuracy* for one searching task.

- Naming: In a naming task, people are asked to name verbally every object they perceive in a stimulus. To measure the naming task, we record the amount of time taken to correctly name an object. Not like the two criterion used in searching target method, only *time* is used in naming measures, because *accuracy* can be too difficult to quantify. The advantage of naming is that the responses of participants are subconscious, so results are less impacted by subjective factors.

- Ratings: A direct way to judge the fidelity of a concerned stimulus is to ask people to give a number which assesses the "amount" of fidelity they see from that stimulus. We call it *ratings* task. The range and scales of numbers measuring fidelity can be freely chosen by participants or assigned by the experimenter. In psychology, empirical evidence has shown that people usually only choose limited values among given responses, such as 7 over 10 possible values for rating in surveys. Thus, participants are usually given at most 7 different rating values to choose in order to reduce overall response variability and limit variability caused by different rating schemes. The values of ratings are collected as the result of experiment for measuring fidelity, and sometimes they need to be normalized before analysis. The advantage of ratings is the simplicity for performing trials and collecting information. The disadvantage is that they are always suspect, because they are a conscious sampling of the subconscious perceptual process. In some of our experiments of evaluating perceptual ability about audio–visual stimuli, for example some participants may give more or even only attention to color or form while others may only notice the movement in a stimulus. Or some participants may attach much importance to sound volume, while others feel that only melody matters. This can introduce a great deal of noise into the ratings. For alleviating the problem, it is required to carefully design question(s) so as to extract the real meaning of participants' responses.

- Threshold: Threshold actually means the limit of perception. The threshold task is therefore used to find the limits of perception beyond which the stimulus can or cannot be perceived. For example, the threshold could be the limit of perception of a noticeable difference between stimuli. The experimenter

chooses a starting stimulus in which the difference is certainly not noticeable to the observer, in other words, the starting stimulus should be estimated far enough from the threshold. By gradually increasing or decreasing the stimulus intensity, the experimenter asks participants to indicate the presence of that difference. One test ends once a participant declares that he/she perceives the difference. Through such a testing procedure, called the "method of limits", we determine the Just Noticeable Difference (JND) threshold. If we allow people themselves to freely adjust the stimulus intensity to achieve the goal, the procedure is called method of adjustment. If the stimulus intensities are randomly varied, it is called method of constant stimuli. If there is a reference to compare with stimulus for each trial, the limit that we will get is *discrimination threshold*. If there is only a stimulus to observe for each trial, we will get the *detection threshold*.

According to specific requirements and research questions, the experimenters can choose one or more experimental measures for the task design. Consider that ratings and thresholds are easily performable (for the process) and interpretable (for the result) for a fidelity measures case once the experimental questions are carefully designed, both of them are adapted in our experiments. Note that the searching and naming tasks are more used in cognitive research than in computer graphics. We will see more details about how to apply these measures for the experimental design in Section 3.2.

3.1.2 Automatic measures

A well designed experiment provides convincing results for fidelity measurement. However, performing an experiment for driving a run-time LOD system is never possible. Instead, we may derive a predictor function from experiment results, and such a function can be used for driving the run–time LOD system. It means that we may consider that the experimental measures provide reliable data for overall automatic measures of fidelity, which should accurately predict experimental results. The good experimental measures can then lead to accurate automatic measures. Consequently, it is important to carefully design and perform the experiment, and finally perform data analysis, in order to have accurate predictor function.

Automatic measures is in fact a method that applies an accepted predictor function in a run–time LOD system. In computer graphics, the automatic measures have

been investigated for static and dynamic imagery, and for 3D models. Intuitively, automatic measures are also feasible for run–time crossmodal LOD rendering if they are supported by good experimental measures. A detailed survey of automatic measures based on image digital form or the results and methods of psychophysical studies can be found in [Luebke et al. 2002].

3.2 Experimental evaluation and psychophysical methods

We have discussed previously the importance of fidelity measures based on perception and the usage of experimental measures used to answer perceptual questions. However, the experimental design is not an obvious and easy task to do but needs a great deal of study on psychophysics, both theoretical and practical. In this section we will review some important methods of perceptual experiments based on psychophysical theories, which therefore serve as standard methodologies for experimental design. With the knowledge of the bias, advantage, and disadvantage of different psychophysical methods in general, we may have a better clue for designing a proper and valid experiment, to answer the proposed research questions.

Psychophysics, when first introduced by Gustav Theodor Fechner in 1860 [Fechner 1860], was a term used to mean a scientific method for studying the relationship between physical world and psychological/perceptual world. Specifically, here psychophysics is used to investigate the relationship between physical stimuli and sensations/perceptions they affect. There is an important literature on psychophysical methods, especially on experimental design, for studying a perceptual system. However, they are mostly for psychologists, rarely for computer scientists. A survey of experimental design for computer scientists can be found in [Cunningham and Wallraven 2011b].

In this section, we introduce at first the procedure of experimental design. And then we introduce the basic psychophysical methods for experimental design by classifying them through the targeted task.

3.2.1 Procedure of experimental design

Perceptual experimental methods need to be carefully designed in order to answer a research question. A well designed experiment usually provides the more usable and more interpretable results. The results can be different kinds, depending on what the research question is and what task participants are asked to do during the experiment.

The research question sometimes can be extremely vague, if we do not know much about what will happen under the given circumstance. For example, the question may be "what will people feel when they are walking in an upside down virtual 3D world", as we have no idea what people will react before performing experiment and we cannot provide any possible answer. On the contrary, when we know what may happen in a given circumstance by means of logical principles from basic assumptions, one hypothesis may need to be formulated before designing an experiment. For example, the precise question may be "How well do people find the distortion artifacts of a 3D object based on an image–based rendering", as we know that people will find the artifacts in such system but we actually want to evaluate their ability. Empirically, the more precise the research question is, the more valid and accurate the experiment and its results are for human visual system to detect them.

From hypothesis to task

Once the hypothesis or research question is formulated, we can consider in the next step what actions should be done during the experiment. According to the potential actions we might apply in the experiment, the reformulation of the research question may be needed. Normally we reformulate the question in order to improve it, so that it can be more explicit. When the question becomes more precise, the number of actions might be less and actions might be simpler or more concrete such as "Yes/No" buttons. It means that reformulation of the question and the design of actions are two procedures of mutual improvement. Once the hypotheses and actions are determined, we may decide an experimental task with selected psychophysical method(s) to apply.

Measurement in psychophysical methods

In psychophysical methods, some provide a variety of descriptions of people opin-

ions which contain a lot of unprocessed information, while others allow quantity measurement. There are two fundamental quantities which psychophysical methods allow to measure: thresholds and scales. Two kinds of thresholds for perception are used in practice: detection/absolute thresholds and discrimination/difference thresholds. Detection thresholds measure the ability of perceiving presence of a stimulus. Difference threshold measures the ability of discriminating between stimuli, and the threshold value is known as Just-Noticeable-Difference (JND), mentioned previously. Scales, on the other hand, describe the level of perception. And they are usually measured by rating related methods, while thresholds are often measured by method of limit and its variants. We will see more details about measurement by means of experiment tasks in the following sections.

3.2.2 Psychophysical Experiment–tasks

What the participants should do or which task(s) should the participants perform during the experiments is the core question in experimental design. There are many possibilities. We give here an overview of the most common tasks that scientists usually use. The psychophysical methods reviewed in the section are classified into different tasks which answer different types of questions and provide different types of answers. Such classification of tasks was proposed by Cunningham and Wallraven [2011b].

3.2.2.1 Research question and answer

The goal of an experiment is to answer a research question. We have mentioned that the more clearly we define the research question, the easier and clearer it is for us to choose the task.

On the other hand, we want to determine the question as specifically as possible to obtain the results as uniquely interpretable as possible. Besides, we want to have the results from which we can conclude the facts for whatever the observed case is. However, it is difficult to design a specific question without certain restrictions on experimental conditions. Therefore we cannot obtain generalized answers under such specific conditions.

The challenge of the experimental design is, as a matter of fact, to find a compromise between specificity and generality. We can see the specificity and generality

of basic tasks in Figure 3.1. In the figure, all tasks are placed on the diagonal of the rectangle coordinates system. The x-axis is the question that a task can answer, ranging from broad/general questions to specific questions. The y-axis is the answer that a task can provide, from concrete to vague. Tasks at one end of the diagonal can answer broad questions, but the obtained answers are difficult to interpret so that one may not draw a concrete conclusion. Tasks at the other end can easily make concrete interpretations, but are only suitable for answering very specific questions.

In the figure, tasks of *broad question* and *vague answer* can be seen as non-measure tasks or qualitative tasks, which concern the quality of stimuli such as free verbal descriptions (see Section 3.2.2.2). Basically, the rating tasks (see Section 3.2.2.3), forced choice tasks (Section 3.2.2.4), and the nonverbal tasks (see Section 3.2.2.5) can be seen as quantitative tasks or measure tasks, in which the participants' actions are used to measure some quantities about perception. And tasks of *specific question* and *concrete answer* are physiological tasks (see Section 3.2.2.6), which are also measure tasks but with respect to the body's natural reaction, such as body temperature, heart rate, etc.

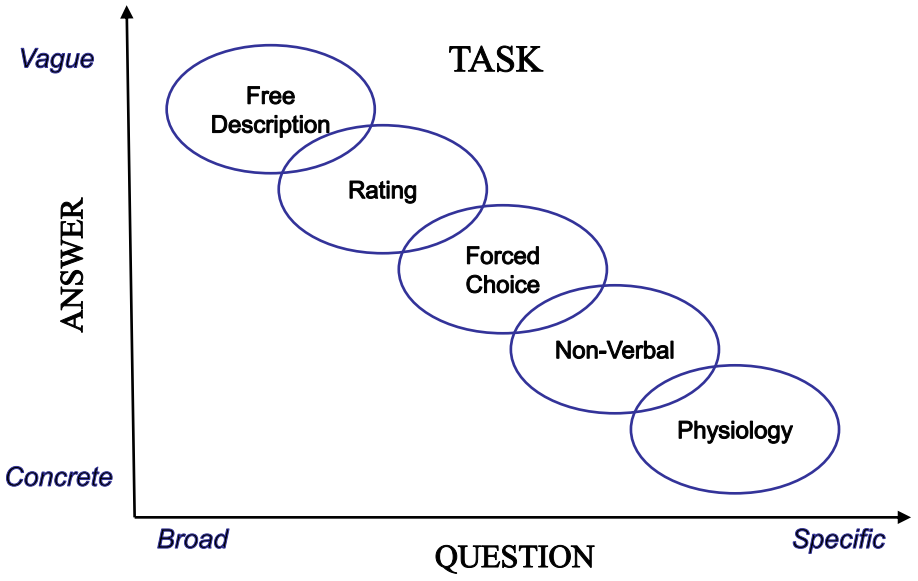


Figure 3.1: Specificity and generality of tasks

3.2.2.2 Free description

The free description task is a qualitative task in which participants are asked to tell their judgments or ideas about what they think or understand from stimuli. In a free-description task, experimenter usually needs to design a set of written questions and ask the participants to write their answers down. Their answers can be very diverse so that information gathered from a free–description task usually provides vague and general answers. Due to this nature of the free–description task, it can be used as a pre-test before a more concrete task that is initially unclear for the experimenter. The pre–test provides relevant answers that hopefully should supply information to guide the design of more concrete experimental task. Such a pre–test may be, for example, the calibration process for further experiments.

The advantage of a free–description task is that it can give a desirable amount of information with respect to experimenter's demands. However, it is hard to give a precise and clear restriction on the type, range, or depth of the information obtained from a free–description. Therefore, the information can be difficult to interpret, and sometimes it might be even irrelevant to the initial research question.

Thus, the free description might not be appropriate for a task that requests clear and direct answers. In other words, a free description task is better for answering a broad, vague and general research question; and if the research question is clear and specific enough, there is no reason to use a free-description task.

3.2.2.3 Rating scale

In a rating-scale task, participants are asked to associate a value with each stimulus according to some specific features, possibly compared with another stimulus (i.e., a reference stimulus) or based on their own criteria. This numerical value, which sometimes might need to be normalized, can be used to determine how the stimulus compares with other stimuli on the dimensions of interest. Thus, different variants of rating-scale tasks provide different results: some provide the ranks of all the stimuli along a dimension of interest, and others provide not only the ranks but also scales, each of which describes precisely how large the difference between two stimuli is. For example, stimulus A is twice as good as stimulus B, stimulus B is four times as good as stimulus C, etc. Note that these responses are most of the time subjective. So, perceived difference along the scale between two stimuli can be very different between participants such that it cannot be considered as an objective

distance between stimuli.

The rating scale tasks include a variety of variants which respond to different demands. Here is the list of common ones which we believe are adaptable to audio–visual perception experiments. More derived forms of rating scale tasks used in computer graphics can be found in [Ferwerda 2008].

Ordered Ranking The simplest variant of rating task is the ordered ranking. In an ordered ranking task, participants are asked to sort the stimuli according to some criteria. The relevant dimension along which the sorting is decided is defined by the experimenter through experiment question or additional explanation. Once the dimension of response is defined, participants are asked to observe all the stimuli simultaneously or sequentially (it depends on stimuli, for example, more than one auditory stimulus cannot be presented to participants simultaneously), and then they are asked to rank the stimuli along this dimension. Sometimes, participants are allowed to freely review any given stimulus. Usually, in an ordered ranking task, if the experimenter provides participants with values to assign to the stimuli, there should be obviously as many values as the number of stimuli to make sure that two stimuli will not have the same value/level. With respect to data analysis, the stimuli are synthetically ranked according to the values they receive.

Magnitude Estimation The magnitude estimation is also a task that requires participants to assign values to stimuli. The difference is that it allows participants to choose any free value along some relevant dimensions without the constraint of a predefined set of values. This task provides not only the ranking, but also the scales between stimuli. However, the value given to stimuli and the scales between stimuli are very subjective and differ a lot among participants, since every one has his/her own standards. Sometimes, data normalization is then needed. Once the relevant dimension is defined, all the stimuli are presented to participants, usually sequentially. Participants assign a value along this dimension to the first stimulus, and based on that they then assign a value to the second stimulus, and so on.

Likert Ratings Similar to magnitude estimation, the Likert ratings task requires participants to assign values to the stimuli. The difference between the two kinds of tasks is that in a Likert ratings task, the set of values is defined by the experimenter. So, participants are only allowed to assign values within the

allowed range. The given scale points can also be labeled, such as "strongly agree", "quite agree", or "slightly agree" for a discrimination test. The stimuli are allowed to "share" a value or label, which means that two or more stimuli can be given the same value or label.

Semantic Differentials The semantic differential task is a transformation of the Likert rating task. In a semantic differential task, the given range of numbers has two ending scale points in opposite signs, such as -2 and 2. Associated labels also reflect this symmetry, and can be, for example, "strongly agree" and "strongly disagree". Between the scale endpoints, we may have as many of such kind of bipolar scales as we need.

Rating scales are quantitative tasks and depend on subjective judgment. The advantage of rating scales is that it is easy to perform the experiment and to arrive at precise and clear answers about scale measures of stimuli. The disadvantage of a rating scale is that it depends on a subjective judgment so that we are not sure whether the answers are reliable enough or if the difference of scales among participants are significant enough to impact/change the results.

3.2.2.4 Forced choice

The forced–choice task is the most common approach in psychological research for perceptual threshold measurements. This approach has a great variety of different forms depending on different demands. The approach is often used in image or graphics experiments, such as face recognition or material comparison.

The forced–choice tasks measure the perceptual limits by asking participants to select one among two stimuli presented according to a given question. Such tasks are well suited for discrimination tests. The essential of the approach is that the forced choice task is in fact a discrimination task. Participants are asked to observe two stimuli (one of them might be a reference) either sequentially or simultaneously and choose one of them based on some criteria. The advantage of the forced choice tasks is that we may construct a perceptual limit function based on the results from such a task, if it is well designed.

For the other general test, a forced–choice task asks participants to choose one answer among a certain number of alternatives according to a research question. For example, participants can be asked to give a description of some aspects of image

stimuli, such as color, shape, material, etc.

According to a research question and some specific requirements, a basic forced–choice task derives a transformed one by adding some restrictions for example. The forced–choice and its derivative strategies are best adapted for a discrimination threshold task.

According to some specific requirements of experimental design, one of the following four forced–choice variant tasks can be selected:

N-Alternative Forced-Choice

In this variant, the experimenter gives N alternative answers and the participants must choose one of them for each stimulus. The same alternative can be chosen by participants for different stimuli. If each alternative answer is hoped to be equally chosen for some reasons, a careful selection of considered stimuli might be required. The alternatives might be direct descriptions, such as "bright", "happy", "fast", or comparative descriptions, with respect to a certain standard such as "faster than normal".

N+1-Alternative Non-Forced-Choice

The N+1-Alternative Non-Forced-Choice is quite similar to the N-Alternative Forced-Choice, except that a last alternative answer allows the participant to refuse to make a choice. For example, the last alternative might be "none of the above" or "I do not know".

N-Interval Forced-Choice

We may consider the N-Interval Forced-Choice as a special variant of the N-alternative forced-choice task. The difference between the two is that in N-Interval Forced-Choice, N stimuli are shown sequentially and the participants are required to associate one stimulus with the required interval based on some criteria, such as "which one of the three stimuli is the fastest".

N+1-interval Non-Forced-Choice

N+1-interval Non-Forced-Choice is similar to the N+1-Alternative Non-Forced-Choice, but derived from N-Interval Forced-Choice. A last alternative answer allows the participant to refuse to make a choice, for example, one alternative might be

"none of the intervals" or "I do not know".

Method of limits

The method of limits is generally used in experiments to determine a threshold of perception by presenting the participants with various experimental stimuli, such as pure tones varying in intensity or lights varying in luminance. It can be seen as a "2–Alternative non forced choice task". The stimuli are presented by increasing or decreasing intensity with respect to certain aspect that has to be detected. If the stimulus is presented from the lowest to the highest intensity, this method is also called *ascending method of limit*, and *descending method of limit* in the reverse case. The participants are asked to declare whether they perceive a difference between each pair of presented stimuli (reference stimulus and test stimulus). The procedure ends once a participant perceives the difference or when all the stimuli are shown.

3.2.2.5 Non–verbal tasks

In some cases, the purposes might be related to a human behavior research. These tasks are called nonverbal or real-world tasks. In Virtual Reality (VR), researchers often use nonverbal tasks to study the behavior in a VR setup, where the perception is strongly tied with behavior. So, the reactions of perception to stimuli can be described and analyzed for behavior or cognitive questions. We can also find such a task in the real world, and it is generally used for cognitive or behavior research. For example, simply asking (by words or by gestures) someone to point to a target, is a non–verbal task.

A special form of the nonverbal task is the method of adjustment previously mentioned (see Section 3.1.1). Participants are asked to adjust the level of a stimulus based on some aspects and stop when the stimulus arrives at some considered criteria. This form is often used for threshold measurements.

3.2.2.6 Physiological tasks

There are some interesting questions that neither the qualitative tasks nor the quantitative tasks can answer directly and well, such as "is the feeling or judgment of participant believable". A questionnaire about participant's background or about their perceived qualities may be used for measuring believability. However, these measures are still indirect and insufficient. Therefore, a physiological task that tests

a natural body's reaction will be more relevant and reliable to answer this kind of questions. Some body reaction measurements have already been invented and widely used in the psychophysiology area. The pulse measurement checks the heart rate; pupillometry tests the pupil dilation; galvanic–skin–response tests the changes on human skins. Such measurements are used in so–called lie detector. The center of gravity measurement is used to measure the human standing center of gravity while a participant is observing a stimulus. Eye tracking measures the rotation of eyes. Functional Magnetic Resonance Imaging measures brain activity by detecting changes in oxygenation of human blood flow and is widely used in clinical medicine. All those measurements can be used as an additional confirmation for quantitative tasks or an independent experimental task, for cognitive research or physiological research.

3.2.2.7 Conclusion

Experimental design is a complicated process which requires deep understanding of psychophysical methods. We have reviewed so far the most common psychophysical tasks based on their measurement content. Finally, we conclude with a grouping tree based on what we want to measure or describe from a task (see Figure 3.2).

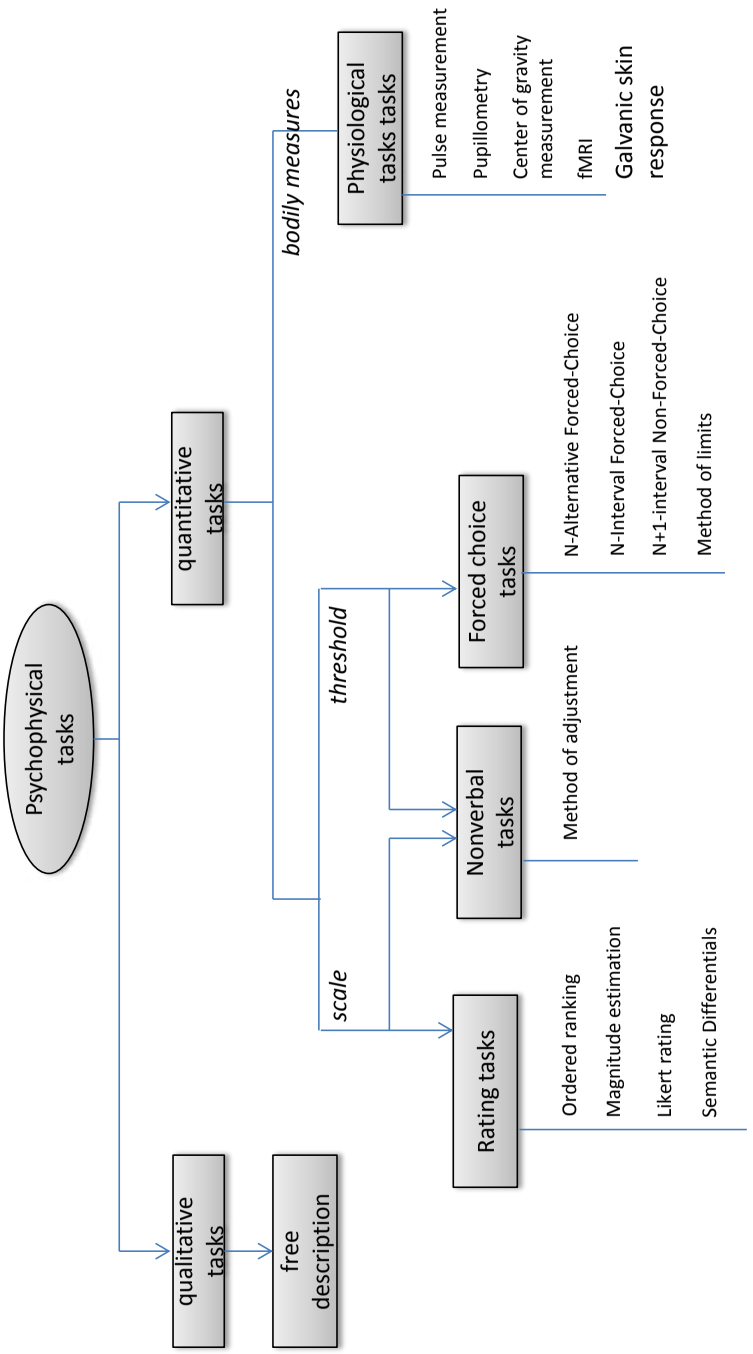


Figure 3.2: Task tree

Part II

Evaluation of crossmodal perception on mono LOD system

Chapter 4

Perception of artifacts in audio–visual LOD rendering

We have raised the research question in Section 1.2.5 that whether and how user perception of artifacts due to impostor–based LOD is affected by a simulated, informative sound. We will attempt to answer in this chapter by assessing the visual ability of artifact detection based on an impostor–based LOD rendering, with and without simulated sound. For doing so, we have developed an impostor–based LOD system in three dimensions which brings discontinuity errors, which are a kind of *artifacts*, and designed a perceptual experiment with a detection task.

The impostor/image–based LOD is a common approach widely used in PC/video games, flight simulation, and so on. The barrier of Image Based Rendering (IBR) to image fidelity is unavoidable because of *artifacts*, such as parallax distortion, popping effects, blurring, and so on. We wonder if the additional modality will impact detection of artifacts, especially caused by impostor–based LOD rendering, which eventually impact the image fidelity adjudged by human visual ability. In this section, we present the implementation of an impostor–based LOD scene, as well as an investigation of perception of parallax distortion based on such an implementation.

4.1 Implementation

We have designed an impostor–based LOD scene with a tree scenario, in which a tree is presented in five levels of detail (LODs) in clipping volume (frustum) according to its location. This image–based LOD system produces the parallax distortion, which we call discontinuity in the thesis, when the billboard–impostors are gener-

ated. Normally during the real–time rendering, the further the point of view is moved away from its ideal position (the one at which the billboards were captured), the larger the discontinuities become.

In real-time applications, the impostor techniques have to deal with the problem of changing appearance of objects due to the changes of viewpoint. The complexity of this process can be reduced by restricting the possible angle(s) of change of viewpoint to a fixed number around a single axis, after applying error measures (see Section 1.1.2) when necessary. When the viewer gets into a position to view the object from an angle close to one of the possible predefined angles, the 3D engine selects the proper billboard and texture and renders the textured impostor at the corresponding location; otherwise, the 3D engine switches to a detailed geometric rendering. The approximate threshold for sorting the current angle in the predefined angles and the limited number of predefined angles can both result in visual artifacts, even when an operation of error measures is added since the process is approximate (compared to perceptual measures). Consequently, the perception of visual artifacts depends on the angle of the virtual camera with respect to the object on which the impostor is applied. To be specific, a larger angle will give a stronger feeling of detection of visual artifacts.

In our implementation, we have applied a real–time impostor–based method with the largest predefined angle (90°) for representing a tree in the perspective volume, since one of the objectives is to investigate for each LOD the smallest change of angle-of-view without impacting the image fidelity. The selection of LOD is therefore controlled not only by the distance between observer and tree (object), but also by the angle of view. Obviously, a larger angle of view for applying an impostor would produce more noticeable visual artifacts. Our first experiment is designed to investigate the thresholds of angles and distances that might be established for a selection method of an impostor-based LOD rendering, with or without an additional modality.

4.1.1 Scenario and graphics rendering

The scenario is a non–animated tree in three dimensions. The tree scene is programmed in OpenGL/C++ for linux, and is generated with an automatically recursive algorithm known as Lindenmayer system (L-system) [Lindenmayer 1975]. Some constant properties of trees, such as the branch level for a tree, the sub-branch

number for each branch, the maximum number of branches, the maximum number of leaves for every branch, ratio length, radius for the current branch of ancestor branch, etc., are provided for the real–time algorithm. The natural hierarchical structure of the tree gives an idea for generating imposed-based LOD, that is to capture snapshots of original details at right places for creating impostors at given LOD, according to the corresponding depth of tree and the distance of tree to viewer.

4.1.2 Impostor–based LOD algorithm

Our LOD algorithm replaces the original detail (rendering) of branches and leaves around certain branch joints (nodes) by billboard impostors of which the size depends on the distance between observer and tree. The impostors are located at chosen branch joints (nodes) so that they replace the branches and leaves around these joints. The distance range for each LOD validation is manually defined for the purpose of simplicity and accuracy. For different LOD, the chosen joints for locating corresponding impostors are different. All joints of the tree are classified by the branch level of a tree, for example, there are 5 levels of depth for the tree in Figure 4.1, consequently there are also 4 levels of joints. For a given LOD, the joints at corresponding level are where we put billboard–impostors. For example, for a given LOD2, we replace the branches and leaves around the first level of joints by impostors.

The number of levels of detail is likely to be less or equal to the number of branch levels of a tree, and in our case we have chosen the same number for branch levels and LODs. Therefore, our tree demo has five levels of hierarchically structural depth making five LODs for representing the tree, which are LOD1 to LOD5. The LOD1 is the graphical (original) detail without impostor replacement. The intermediate LODs (from LOD2 to LOD4) are the partial graphical detail combined with impostors replacing branches of corresponding depth. LOD5 is one impostor replacing the whole tree. Examples of GLOD2 to GLOD5 are shown in Figure 4.2 where we illustrate the screenshots of the tree in different LODs used in experiment (to be introduced later in Section 4.2). Besides, we can see two examples of a simple tree in LOD2 and LOD3 with a good clarity in Figure 4.3 in which we illustrate impostors by adding white frames.

For the purpose of perceptual evaluation of an audio–graphical scene, we focus on the rendering of a tree that is appropriate for dealing with both LODs (because

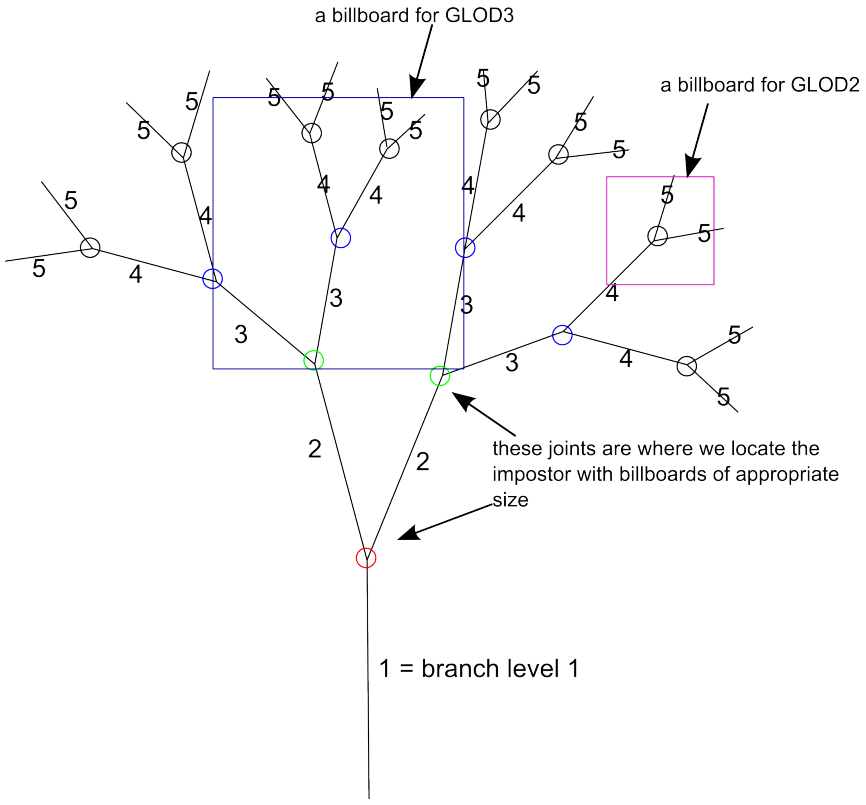


Figure 4.1: Illustration of the impostor method applied in tree scene.

of its fine and complex structure) and audio rendering (because of the possibility to render the noise in a tree caused by wind). Our tree model is made of textured truncated cones (trunks and branches) and point–sprite primitives (leaves).

As previously mentioned, some constant parameters of tree are provided for running the tree LOD algorithms in real time, which are the branch levels of a tree, the sub-branch number of each branch, the maximum number of branches, the maximum number of leaves of each branch, and the ratio length and radius of the current branch of ancestor branch. Since the tree is structurally hierarchical, this gives us a hint for generating LODs. Specifically, we apply real–time impostors according to the structural depths of tree and the distance between the tree and the observer.

In detail, our LOD algorithm replaces the current representation of the tree by impostors captured on–the–fly from the original detail of the tree. As mentioned previously, one captured impostor should be applied at the appropriate joint of a

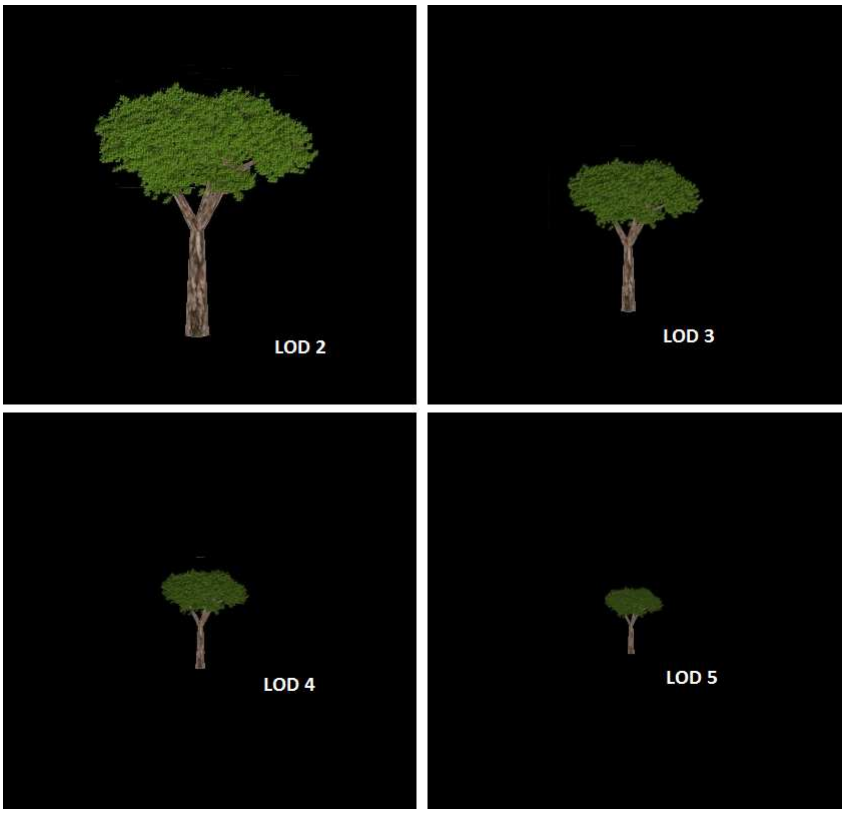


Figure 4.2: Tree of GLOD2 to GLOD5

tree. And the zone of the tree that needs to be replaced depends on the current LOD level and branch levels (depths). In our case, the replacing zone of the tree is the parts of corresponding branch levels for a given LOD. This means that for LOD5, the impostor should be captured from the first branch/trunk level to the last level (which means the entire tree); for LOD4, it should be captured from the second branch level of the tree to the last level; similarly for the other LODs (see Figure 4.1). Hence, there are as many LODs as the branch levels. In other words, the tree has five hierarchical branch levels, while the LODs for the tree based on distance is also five, from LOD1 to LOD5. Naturally, the distance range for each LOD is manually set by segmenting the scene zone into five linearly.

For supporting navigability, the camera is allowed to move around in the scene so that the tree can be seen from different points of view and distances. When the camera (the observer) is moving forward, the immobile objects in the scene seems to be moving to the camera (the observer). Notice that in some places of the thesis, we say that "the tree is moving" instead of "the camera is moving" for the simplicity

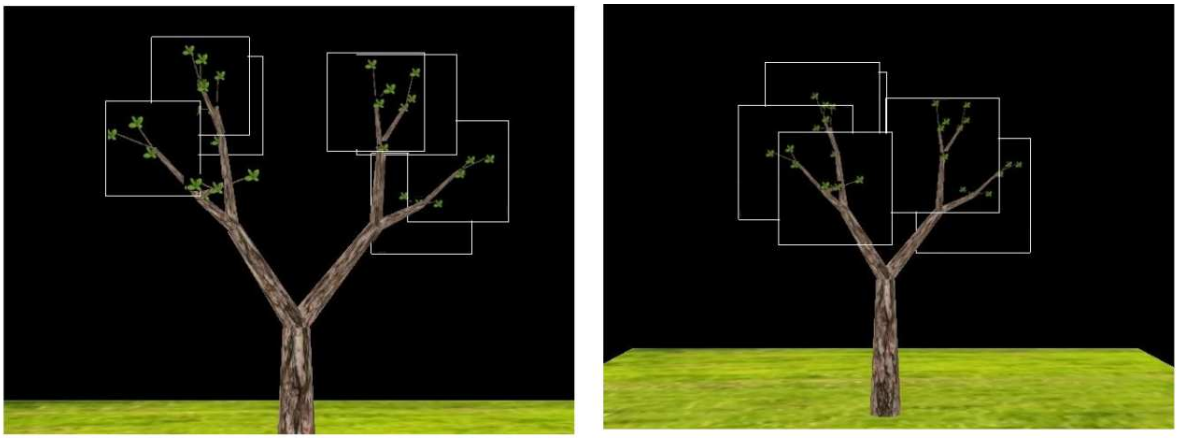


Figure 4.3: The left snapshot is the LOD2 and the right snapshot is the LOD3. Impostors are marked by white frames

in algorithm description.

To select the appropriate LOD for the tree at every step/frame as the camera moves, the *distance* is used as the selection factor. The maximum range of the clipping volume (between frustum near and far) can be segmented into different LOD selection along the z-axis, as mentioned previously. In our case, the view range along z-axis is 200 and is manually segmented into five segments: from 0 to 25, from 25 to 50, from 50 to 80, from 80 to 130, and from 130 to 200. The tree is represented in LOD1 (original detail) when it is located in the segment 0-25, in LOD2 when it is located in the segment 25-50, and so on.

Once the tree moves from the current LOD into a lower LOD at the first step, a full detailed rendering will be performed at the initial time. At the same moment, a screenshot of the scene is stored in memory, and is prepared for generating the corresponding textures (of impostors) to be used in next frames if the tree stays in the "lower" LOD (which should be actually the current LOD). After the initial time, if the tree stays in the "lower" LOD, we render quadrilateral textured–impostors based on the screenshot to replace corresponding parts of the tree in original detail, according to the LOD level and branch levels as mentioned previously. This process is how we produce in real time the impostors, and such impostors are used as long as the tree stays in the same LOD range. We can see in Figure 4.4 how the tree is rendered from LOD3 to LOD4. In this figure, the tree is moving from time $t-1$, then to time t, and finally to time $t+1$. At time $t-1$, the tree is located in the LOD3 area and rendered in impostors (the second branch level to the last branch

level) and geometric detail (the trunk and the first branch level). At the time t, the tree is initially entered into the LOD4 range, so it is completely rendered in geometric (original) detail and then the tree scene is stored as a screenshot. When tree is moving into time $t + 1$, only the trunk is rendered in geometric detail, while one quadrilateral textured–impostor tailored from the screenshot is used to replace the geometric detail from the second branch level to the last branch level.

In the reverse case, which is when the tree moves from the current LOD into a higher LOD at the first step, a geometric detail rendering for the whole tree is performed at the initial time the same as in the case aforementioned . But then, instead of taking snapshots from the current screen, we allocate an offscreen buffer to render the tree in the full detail at the minimum distance that defines the higher LOD range (see Figure 4.3). New snapshots are stored in the offscreen buffer to generate the impostors at appropriate locations and with appropriate size, as long as this higher LOD is valid (see Algorithm1).

```
foreach camera step do
    if LOD=LOD1 then
        Render in full detail at screen;
    else if camera moves forward and LOD=LOD2/LOD3/LOD4/LOD5
    then
        if tree goes from lower LOD to a higher LOD then
            Render in full detail at screen at higher LOD;
            Capture and record snapshots impostors in memory;
        else
            Render in impostors representation;
        end
    else if camera moves backward and LOD=LOD2/LOD3/LOD4/LOD5
    then
        if tree goes from higher LOD to a lower LOD then
            Render in full detail at screen at lower LOD;
            Render in full detail at an offscreen buffer from which we take
            snapshots impostors in memory;
        else
            Render in impostors representation;
        end
    end
end
```

Algorithm 1: Algorithm of LOD selection

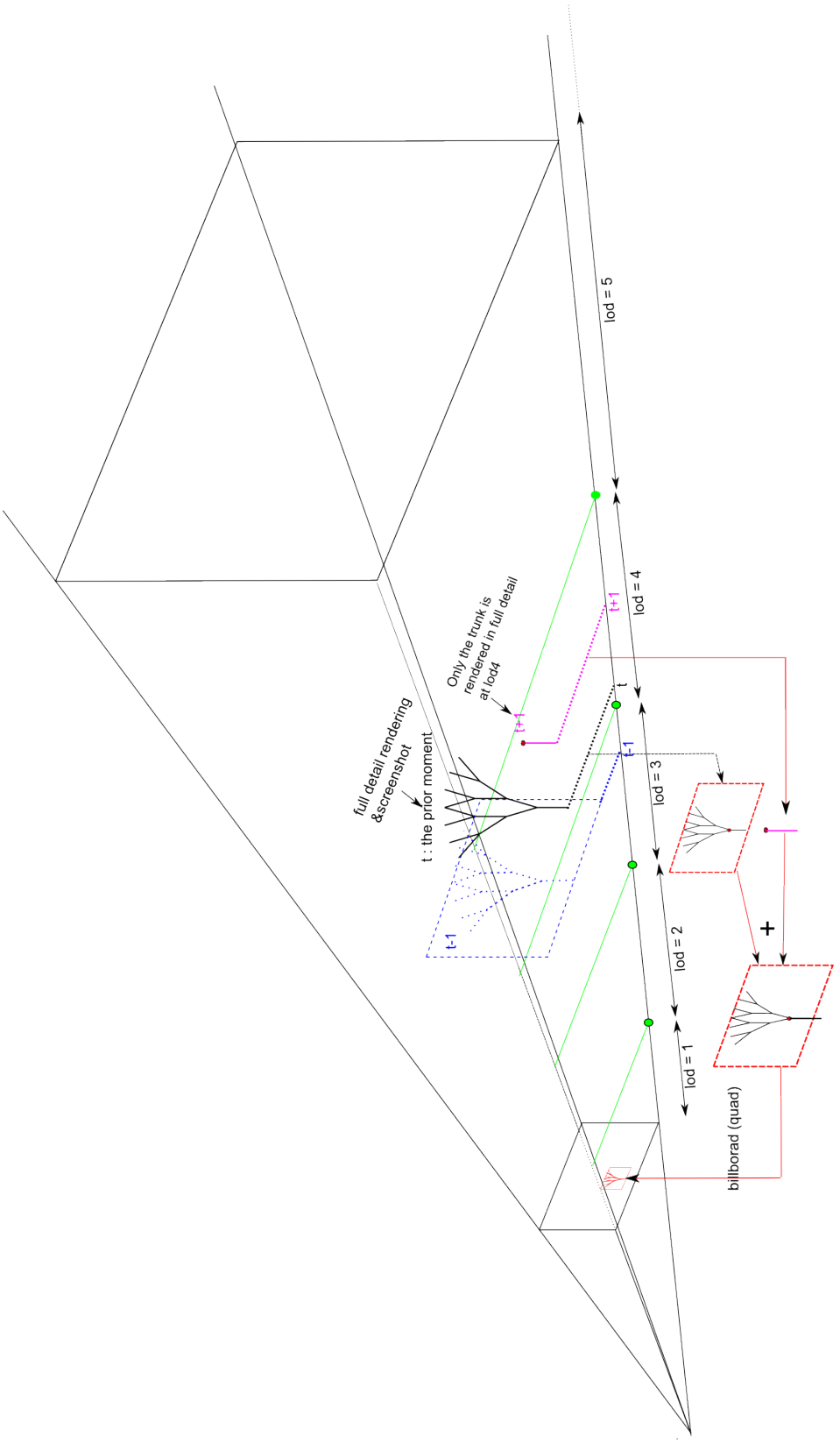


Figure 4.4: Illustration of impostor–based tree rendering from LOD3 to LOD4.

As a result, each time when the tree moves from one LOD to another, there is one screenshot made from full–detail scene, for generating textures used for generating quadrilateral billboard impostors in the same LOD range. Intuitively, it makes the discontinuity artifacts less noticeable when the angle of view is small and the tree is far away from the viewer. Consequently, the parameters for defining an impostor such as the distance and the angle of view should be the selection factors for managing LODs in real time. So, the thresholds of factors that determine the detection ability of artifacts are worth of investigating, normally through empirical evidence. Consequently, we investigate the impact factors of detection ability of artifacts by performing a perceptual experiment on human perception evaluation, which is presented in the next section.

4.2 Experiment design

The discontinuities are the main visual artifacts arising from our impostor–based LOD system, and our purpose is to investigate whether an auditory modality may impact the human perception ability of artifacts detection. As a result, the objective of the perceptual experiment is to evaluate the perception of visual artifacts on simple– and bi–modality. The stimuli are the images captured from tree scene with or without a simulated sound of the tree in the wind.

The research questions that have been raised before the experimental design are:

1. How do observers perceive the artifacts arising from our impostor based LOD?
2. Does sound make a difference on the perception of image artifacts (discontinuities)?
3. Does auditory modality reduce or emphasize the critiques on the perception of artifacts (discontinuities)?

Based on the above questions, we have designed a series of experiments.

4.2.1 Auditory integration

We are interested to understand the influences that a role sound might have with respect to visual fidelity. The question has been open for decades. Recent

work has studied how spatialized sound in 3D graphical environment can enhance and reinforce the visual comprehension of the scene and some experiments have shown that accurate sound modeling gives users stronger realistic feeling in virtual visual environments [Larsson 2002; O'Brien et al. 2002] and high-quality sound enhances perceived visual quality [Grelaud et al. 2009]. Some work uses synthesized sound on the fly [O'Brien et al. 2002; van den Doel and Pai 2003], recorded sound, or a combination of them [Bonneel et al. 2008; Grelaud et al. 2009] to simulate a physical animation, e.g., bowls fall on the ground, vibration sound from a wood bar. Some acoustic work simulates geometrical propagation paths of physical phenomenon in order to give a stronger sense of presence in virtual environment [Funkhouser et al. 1999; Tsingos et al. 2001].

However, since our tree scene is generated in a non–animated context, we consider neither the realistic sound rendering such as spatializing and propagation, nor the sophisticated sound synthesis for a sound event. We simply combine a recorded sound (a sound simulating a tree in the wind) with graphically static tree. Only the sound sample and its impact on the perceived image quality are considered in our case. As a result, we have generated a stereo sound for our visual scene. Its volume is tuned based on the distance between tree and observer. For the navigable scene, the 3D statistic graphical tree will be accompanied with a continuous stereo sound of a tree in the wind, and the stereo sound reduces or reinforces the volume as the viewer moves away or close to the tree. For our perceptual experiment, an auditory–visual stimulus is the image captured from our tree scene accompanied by a simulated sound at a corresponding volume based on the location of tree.

4.2.2 User interface

We have designed a self–documenting User Interface (UI) to assist participants in performing the experiment (see Figure 4.5). In the left part of the UI, a stimulus is presented; on the right hand side, the answers are displayed together with an information area . During the experiment, the participants are not disturbed by experimenter, and the UI allows them to control progress of the experiment.

4.2.3 Participants and apparatus

Twenty subjects from different disciplines, aging from 18 to 40, have participated the experiment. They were invited in an anechoic (sound proof) room, sitting 0.5

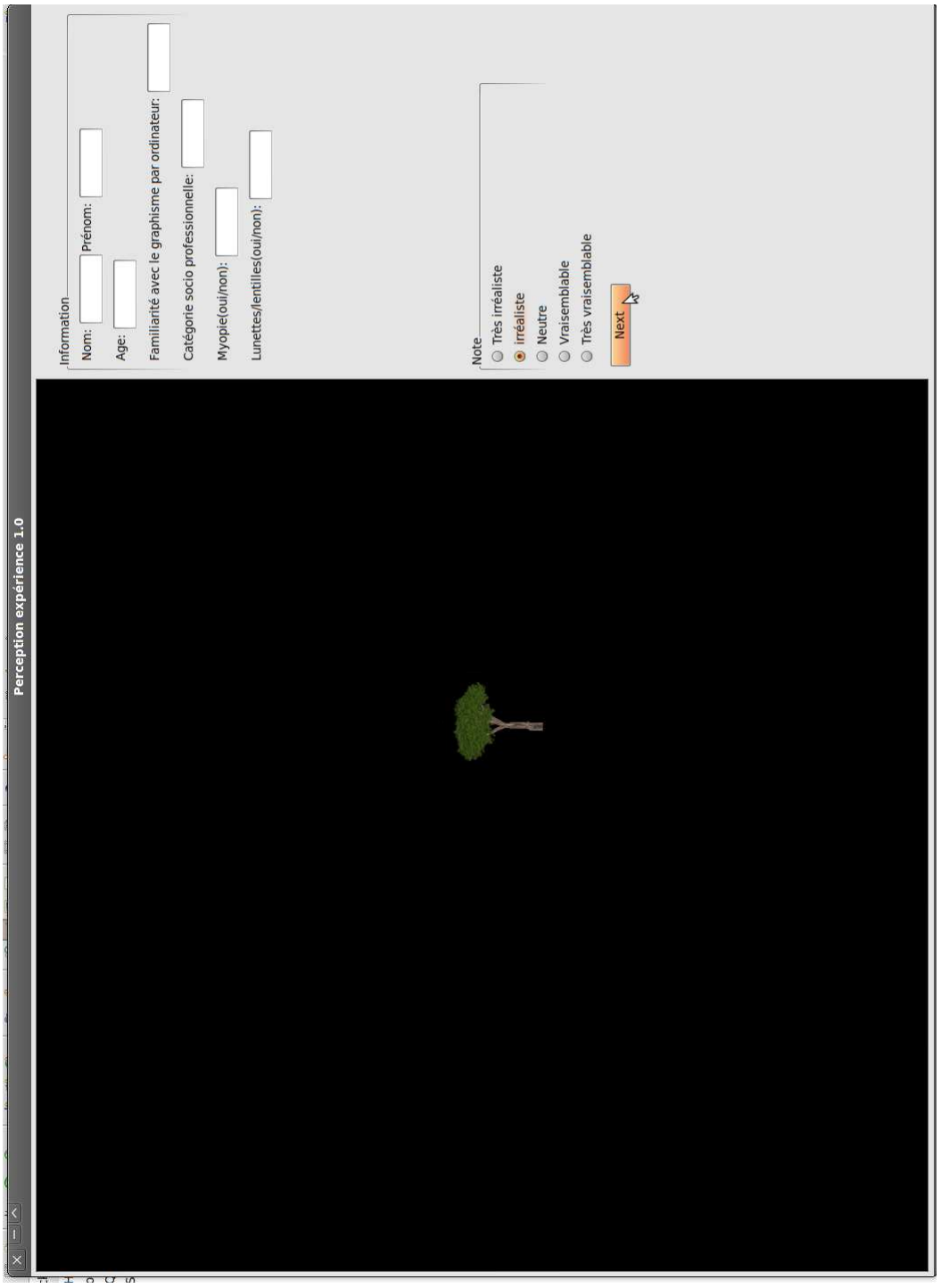


Figure 4.5: The user interface for the experiment about sound influences on impostor–based LOD

meter away from the monitor, equipped with binaural loudspeakers. All of the participants have good hearing and good or corrected vision. In the beginning of each experiment, each participant enters his/her personal information through the UI.

4.2.4 Stimuli and procedures

The question we had asked participants to answer is "Do you find the tree scene in the image realistic?" and they had to choose one of five possible answers which are (i) very realistic, (ii) realistic, (iii) neutral, (iv) unrealistic, and (v) very realistic. Before performing the experiment, participants were explained the definition of "realistic" by illustrating them the "realistic" tree which is an image of full detail tree and the "unrealistic" tree which is an image of *impostored* tree with visible artifacts. They were shown with a series of images captured from the tree scene with or without simulated sound, and answered the question for each stimulus. The scene images were captured at different given points of view and locations/distances, and at each location and point of view we captured one image with impostor rendering and one without (that means full detail rendering). Each image was shown with and without simulated sound, whereas all images were shown in random order.

Since participants were asked to give a labeled value on each stimulus about some concerned aspects, specifically the realism, based on common sense established before experiment, the experiment task that we chose to perform is considered as the *Rating Scales* task, more explicitly *Likert Ratings* (see Section 3.2.2.3).

Twenty people were invited to perform the same experiment in which each of them was shown with two series of stimuli: one with sound and the other without sound. The snapshots/images were captured from the rendering of tree scene at different LODs varying from LOD2 to LOD5. The snapshots were captured with about 10 different angles of view for each LOD from LOD2 to LOD5, resulting in 40 positions in total. At each position, two snapshots were taken for tree scene: one is full–detail graphics and the other is impostor–based rendering. Therefore, 80 snapshots without sound were shown for the first series of stimuli. The first series of stimuli with and without the impostor rendering are selected randomly, and we investigate whether and when the participants distinguish the impostor and full–detail rendering by evaluating the thresholds on the perception of visual artifact. The second series of stimuli were the same as the first series, but they were accompanied

by stereo sound simulation with different volume based on the distance between the tree and the camera. Participants were allowed to freely observe the stimuli and declared how much the tree seems realistic or unrealistic to them (which means, specially in our case, whether or not the participant will notice the visual artifacts) by choosing one of the five labeled answers.

We have manually controlled the selection process of the snapshots of rendered images. For each LOD from LOD2 to LOD5, we select the snapshots with different angles of view for the tree but with the same distance from tree to x-axis. Since the view range is divided into: (i) <25 (full detail), (ii) 25-50 (LOD2), (iii) 50-80 (LOD3), (iv) 80-130 (LOD4), and (v) 130-200 (LOD5), we choose 36 as D2 the distance from tree to axis X for LOD2, 59 as D3 the distance from tree to x-axis for LOD3, 91 as D4 the distance from tree to x-axis for LOD4, and 135 as D5 the distance from tree to x-axis for LOD5, respectively.

Note that our field-of-view angle is 60°. For D2, we select 12 snapshots of angles of view from 0° to 14° (we have filtered out the angles of view >15°, because the artifacts are simply too obvious). For D3, we select 12 angles of view from 0° to 6°, for the same reason of LOD2 for the filter of angles. For D4, we select 8 angles of view from 0° to 12° (with the same reason of LOD2 for the filter of angles). For D5, we select 8 angles of view from 0° to 30° (no filter on angles, because there is no obvious artifacts).

The answers provided by participants are recorded as an integer number from 1 to 5, presenting from very realistic to very unrealistic, in order to statistically analyze the experimental results.

4.3 Statistical analysis

We apply the statistical method, analysis of variance (ANOVA), for analyzing the obtained data to evaluate the similarity between impostor-based rendering and full–detail rendering and the between impostor-based renderings with and without sound. ANOVA is a statistical tool for studying the significance of group means by analyzing the variance between groups. Here, we are interested in the two metrics: (i) the similarity between impostor-based rendering and full–detail rendering and (ii) the similarity between impostor-based renderings with sound and without sound, and will perform the evaluation in the following.

4.3.1 Evaluation of visual perception of artifact

We perform ANOVA on data from the same angle of view to evaluate similarity between impostor rendering and reference rendering (full–detail rendering) as within–subject factors. Note that if *P–value* of ANOVA approaches to 1, the similarity result between impostor-based rendering (LOD) and full–detail rendering (noLOD) is significant.

Through ANOVA, we get the table of *P–value* on every angle of view of impostor. The results are summarized below:

1. For LOD2, $P > 0.9$ when the angle of view is smaller than approx. 7°;
2. For LOD3, $P > 0.9$ when angle of view is smaller than approx. 2°;
3. For LOD4, $P > 0.9$ when angle of view is smaller than approx. 5°;
4. For LOD5, $P = 1.0$ for all the angle of view.

The above results show that firstly participants cannot notice the difference between LOD and noLOD in LOD5 which means that they can hardly observe the visual artifacts when an impostor is replacing the whole tree. As a supplementary information, we plot (see Figure 4.6) the average scores for all the snapshots of each distance from tree to x-axis. In Figure 4.6, the red dotted line shows the average score of rendering with impostor for every distance from tree to x-axis (LOD) while the blue line indicates the average score of full–detail rendering for every LOD distance from tree to x-axis (noLOD). Dist1 is the distance from tree to x-axis for LOD2; Dist2 is the distance from tree to x-axis for LOD3; Dist3 is the distance from tree to x-axis for LOD4; Dist4 is the distance from tree to x-axis for LOD5. We can see that, at Dist4 (i.e., the LOD5 distance), the average scores of LOD and noLOD are very close. However, at LOD2, LOD3 and LOD4, the average scores are very different due to the impact of perceived image artifacts.

Secondly, for LOD2, LOD3 and LOD4, subjects notice the visual artifact when the angle of view exceeds a threshold: 7° for LOD2, 2° for LOD3, and 5° for LOD4, respectively. These thresholds obtained by the analysis of perception can be used as reference to predefined angles at which LOD switch must occur. This could be very interesting because our LOD selection algorithm could be consequently improved by rendering full–detail and recapturing the screenshot for textured–impostors when the angle goes beyond the threshold from last impostor.

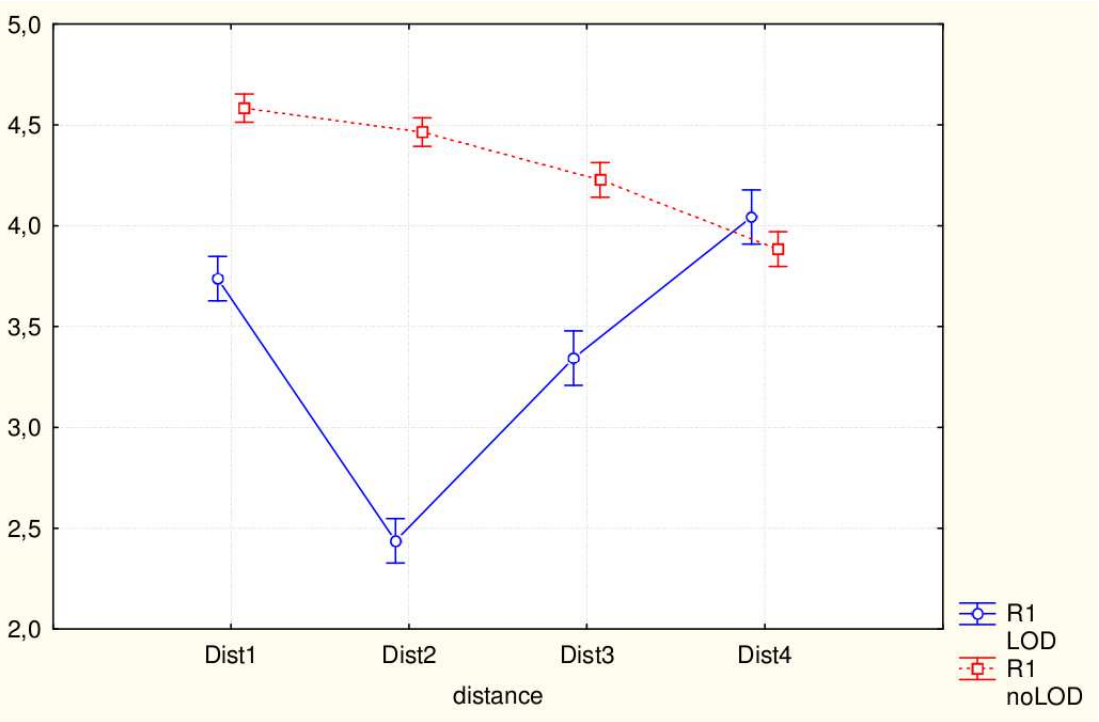


Figure 4.6: The average scores of LOD5 and noLOD5 are very similar.

The reason that the threshold for LOD3 is even larger than the ones for LOD2 and LOD4 could be that the discontinuity artifacts are be less noticed when the impostor is applied on the area which has a high spatial frequency such as tree leaves at LOD2. This is understandable, because according to Contrast Sensitivity, people perceive less contrast at high spatial frequency. On the other hand, in the area which has a relatively low spatial frequency such as trunk area at LOD2 or LOD4, people perceive more contrast (see Figure 4.7).

Thirdly, it is clear that under a certain threshold of angle of view and distance, the artifacts of discontinuity-type are not noticeable by human vision. Furthermore, one can find that the two parameters can be manipulated for controlling LOD selection in real time rendering.

4.3.2 Evaluation of perception of visual artifacts with auditory effect

We also apply ANOVA for analyzing the similarity between impostor-based renderings with sound and without sound. Different to the study on impostor-based

Figure 4.7: The visual contrast on LOD2 and LOD4

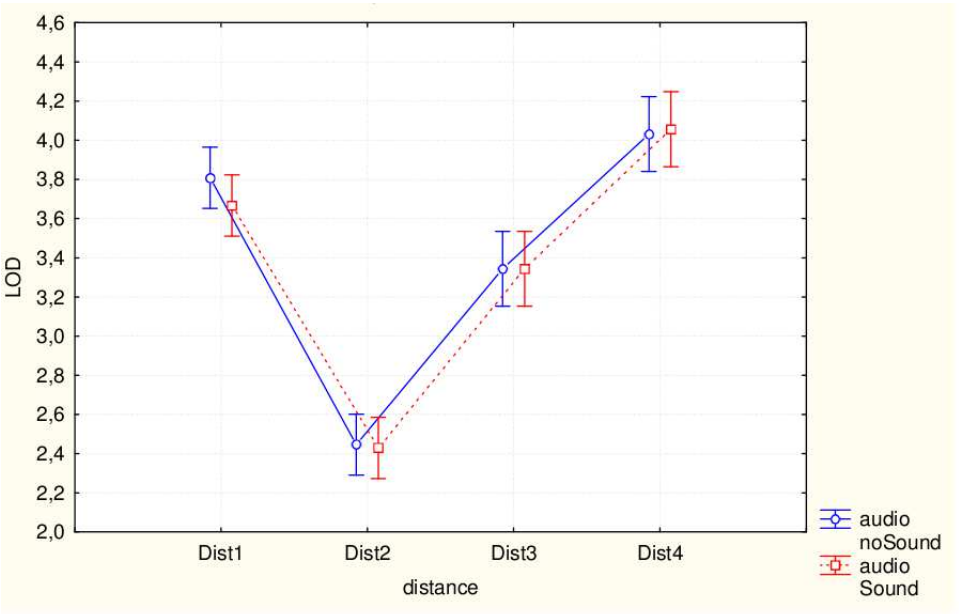
The left snapshot is captured from LOD2, for which the snapshot impostors are basically pasted onto the leaves area. The leaves area has complex information that confuses our perception of visual artifact. The right snapshot is captured from LOD3, for which the snapshot impostors are pasted onto the leaves and trunk area. Since the trunk area has much simpler information than leaves area, one may easily observe the artifacts on the trunk area. Note that we do not consider the effect of aliasing as a visual artifact.

rendering and full detail rendering, we perform ANOVA one time for all data in each group instead of regrouping them by angles of viewpoint. Note that here the data of full detail rendering are not taken into account for analysis.

It is found that the P-value of ANOVA is about 0.76172 between impostor-based renderings with sound and without sound, which means in general subjects do not perceive great difference between the snapshot of impostors with and without sound. In other words, the stereo sound simulation does not obviously impact the perception of visual artifacts.

We also provide a supplementary figure showing the average scores of snapshots with and without sound (see Figure 4.8). From this figure, we can see that the average scores slightly decrease from snapshots with sound to snapshots without sound at Dist1 and Dist2 (the distance from tree to x-axis for LOD2 and LOD3), and meanwhile slightly increase from snapshots with sound to snapshots without sound at Dist4 (the distance from tree to x-axis for LOD5). Given by the first analysis on LOD2 and LOD3 in the previous section, we know that the stereo sound will somehow slightly aggravate the feeling of perception of visual artifacts; conversely,

Figure 4.8: The average scores for every LOD with sound and without sound.



The red dotted line indicates the scores of snapshots with sound while the blue line indicates the scores of snapshots without sound.

users do not notice the visual artifacts caused by impostors at all for the LOD5. Therefore, sound simulation slightly improve the quality of visual perception.

Result
We find that the simple stereo sound simulation can only give a more realistic feeling for visual perception when no visual artifact is perceived. But the sound slightly aggravates the feeling when there are perceived discontinuity-type artifacts.

4.4 Conclusion and perspectives

The main objective of the evaluation of impostor–based LOD system is to investigate the impact of audio modality on visual perception/detection of artifacts arising from impostor–based LOD rendering. The experiment shows that simple stereo sound can hardly impact the perception of image discontinuity artifacts. However,

there is a tendency that sound enhances the visual perception when there is no image artifact perceived. On the other hand, sound can slightly aggravate the perception sense of defects when the image artifacts have been noticed. This conclusion implies that during impostor–based LOD selection, representation of objects should be less degraded when a simulated sound of scene is added, compared to the case when there is no sound. It contradicts the conclusion of [Bonneel et al. 2010], due to the fact that they focused on the perception of materials for which the LOD method they have applied does not produce visible defects. We however still confirm that sound enhances the perceived image quality when there is no visible artifact. In the future, we will consider different type of artifacts and integrate graphical rendering with a realistic spatialized sound.

Part III

Audio–visual description and modeling

Chapter 5

Spatial audio–graphic modeling for X3D (see Appendix A [Ding et al. 2011])

In this chapter, we discuss our work of *spatial audio–graphic modeling in X3D* (see Appendix A [Ding et al. 2011]), which offers a new way of modeling audio–graphic content for interactive 3D scenes with the concept of *sound processes* and their activation through 1D, 2D or 3D profiles and extends X3D standard to represent the sound process and activation profile model for providing a rich audio–graphic description.

We have argued that, in audio–graphic scenes, the visual and auditory modalities are synchronized in time and space, so that the corresponding synchronous events may be determined by an audio–visual process. As a result, audio–graphic scenes may be modeled by a parametric process with respect to bimodal events. We believe that the modeling method should be represented in a standard file format for encoding audio–graphics scenes. In our collaboration with partners from different backgrounds (under ANR project *Topophonie*), we have found that such a file format is needed in computer graphics, in VR, and even in sound synthesis for sharing audio–graphic scenes between different softwares and between different systems.

As a matter of fact, today's 3D applications, such as PC/video games, often integrate high–quality graphical and audio effects to provide a more realistic user experience. However, practicers mostly render graphics and sound separately. A common model and related interchange format for interactive audio–graphic scenes is lacking, to the best of our knowledge.

In [Ding et al. 2011], we explored a new way of audio–graphic scenes modeling based on a novel concept of *sound process*, and then represented the resulting audio–graphic content in X3D format by extending its initial features.

Note that in Section 6.1.2, we have discussed the concept of *sound process* and *activation profile* for generating the audio–graphic scene of the "tree in the wind" example. In the following, we will discuss how we model audio–graphic content of interactive 3D scenes with the concept of sound processes and their activations through 1D, 2D or 3D parametric profiles and represent the audio–graphic content in an extended 3D format (X3D).

5.1 Background and motivation

In the context of 3D graphic scenes, researchers have long recognized that high–quality sound can improve the user experience in a virtual environment. Much work has been done on the method of sound spatialization in 3D graphic environment. For instance, O'Brien et al. [2002] and Bonneel et al. [2008] focus on sound synthesis for diverse physical motions, while Funkhouser et al. [1999] and Tsingos et al. [2001] compute the sound propagation paths by simulating acoustic phenomena. People also investigate computational optimization by using auditory culling and clustering techniques for reducing the complexity of audio rendering. However, none of these works studies the issue of sound mapping and activations in a 3D graphic environment. We are thus motivated to address this topic by using the concept of *sound process*. By specifying the activation procedure of *sound process* with specific characters, we can establish a common modeling method with respect to audio–graphic scenes.

It is clear that although the concept of *sound process* is adaptable to the interpretation of most audio–graphic events/scenes, a standard file format is required for representing the context in order to interchange data between different platforms. A multi–modal modeling method supported by a standard file format for sharing and exchanging data is essential. Instead of creating a new format, we therefore consider the widely used markup language for 3D representation, X3D. Notice that X3D supports spatial audio representation and almost all basic and advanced 3D techniques in computer graphics. However, it is worth pointing out that the spatialized sound in X3D is still for basic use and is not refined enough for representing an interactive audio–graphic scene in complex context. Therefore, we consider to

incorporate the concept of *sound process* and its activation procedures for modeling audio–graphic scenes in XML–based X3D format by adding a new extension schema to the conventional one. In practice, the extended X3D format is well suited to represent complex audio–graphic modeling. Note that thanks to X3D, our method is able to be standardized for more general usages and is extensible for any future feature.

In the coming Section 5.2, we will discuss some technical knowledge about X3D. In Section 5.2.2 we will introduce the XML schema language that we decide to use for describing our extended structure of X3D document. In Section 5.3, we will summarize the principles of our 3D audio–graphic scene modeling method based on *sound process* and *activation profiles*, which is published in [Ding et al. 2011]. Section 5.4 discusses how we represent our new audio–graphic modeling method in extended X3D through XML Schema, for the completeness. In Section 5.5, we offer a guideline of modeling audio–graphic scenes in extended X3D through an event chain example. Finally, Section 5.6 gives the conclusion.

5.2 Technical points about X3D

We will discuss some technical knowledge about X3D in this section so as to have a better understanding on why we choose X3D for representing our modeling method and how we design the representation through this 3D format.

5.2.1 Related formalism

Among all the other 3D formats, there is one very similar to X3D, which is also based on XML schema and able to represent rich 3D content, called COLLADA.

Both X3D and COLLADA support 3D graphics in an advanced level. However, their applications and targets can be very different. X3D focuses on the visualization of 3D assets within applications, while COLLADA focuses on the pipeline tool that exports 3D content and assets from diverse modeling tools to an application. Note that COLLADA is mainly used in the game industry.

So, X3D is more open to the future features that may used in a 3D visuallization. Note that X3D especially supports spatialized audio, i.e., the audio–visual sources can be mapped onto geometry in the 3D scene. Moreover, the extensionality of X3D

allow us to integrate our concept of *sound process* modeling in X3D standardization, for representing the 3D audio–graphic scenes.

5.2.2 XML Schema

An XML schema is a description of constraints on the structure and the content of an XML document such as X3D. There are several languages available for specifying an XML schema, one of which is defined by World Wide Web Consortium[1] (W3C), called XML Schema, also known as XSD (XML Schema Definition). Note that in this thesis, *XML Schema* refers to the language of XML schema defined by W3C, while *XML schema* refers to any XML schema language.

XML Schema is a successor of DTD, which is one of the traditional formalism to express constraints on an XML language. W3C policy is to replace DTD by XML Schemas, because they are (i) written in XML, (ii) extensible to any possible features, and (iii) able to support XML namespaces. XML Schema is considered to be a richer and more powerful XML constraint language. To specify the rules and structure of our audio–graphic modeling in X3D, the XML schema of X3D needs to be extended. In Section 5.4, we will see the design of the structure of the extended X3D, while the extended XML Schema can be found in Appendix B.

5.3 Audio–graphic modeling principles

In an audio–graphic scene, graphics usually contain visual information, while sound often delivers auditive information that is relatively invisible and perceived only when activated. Graphic and sounding objects are audio–graphic, when visual and audio modalities are synchronized in time and space and when they share a common process. Based on this point of view, we focus on the general method for placing the sound sources in a 3D graphic scene and mapping them into *sound processes* that are activated by parametric activation profiles when related audio–graphic objects or events occur.

The method is determined as follows:

- Each sound *effector*, or often called sound source, is situated at a single point

[1] W3C is the main international standardization organization for the World Wide Web. `http://www.w3.org/`

of a geometry object in the 3D scene. Such a point is called sound point.

- Each sound point is linked to a *sound process*. Many sound points may be assigned to the same *sound process* model, but they have individual *sound process* instance.
- The activation level of a *sound process* is measured at an assigned sound point.
- The *activation* of *sound processes* and related parameters are determined by *profiles* or *maps* that move in the 3D scene.

Take a sound made by leaves in the wind as an example that we have seen in the implementation of scene in the previous chapter. Every leaf, or a unit of a group of leaves, is supposed to be a sound point which is linked to a leaf *sound process* model, and the wind is considered to be a kind of *activation profile*, represented as a geometry object for example. When the wind profile moves through the leaves, the leaf *sound processes* will be activated according to a kind of interaction between them. We argue that such sound processes can be applied in most cases of audio–graphic scenes.

The related concepts are briefly provided as follows:

Sound process :

Based on geometry information of audio–graphic objects, sound processes are mapped in a 3D audio–visual environment. They are activated by certain parametric profile when corresponding audio–graphic objects occur. A *sound process* is located at one point or many points in a group.

We define the *sound process* by three general parameters, *identifier*, *model*, and *activations*. (i) *identifier* is a string type of parameter that identifies an instance of a *sound process*. (ii) *model* is a string type of parameter, and it refers to the class of the *sound process* running the instance. Such classes can be leaf *sound process*, traffic *sound process*, rain *sound process* and so on. (iii) *activations* is a vector parameter which contains a set of numbers between 0 to 1. These values determine the activation levels of the *sound process*. They are general and applicable for most of the *sound processes* implementations. As a result, we consider them as basic parameters for *sound processes*. Based on the basic parameters, we specify additional concrete parameters for individual *sound process* models. For instance, we may add an integer type of parameter

namely *impact material* for rain *sound process* model, or add a floating–number type of parameter namely *car speed* for traffic *sound process* model.

Note that the volume of the *sound process* is tuned by its individual sound placement.

Sound sources and placement :

We assign *sound processes* to sound points in 3D scenes, e.g., we assign leaf *sound processes* to leaves, or we assign rain *sound processes* to raindrops. A sound source should be located at a sound point so as to associate itself with a 3D primitive or object. So, for a concrete scenario, one or more sound sources should be linked to the same *sound process* model in order to emit sound when the *sound processes* are activated by a certain profile. For example, all the raindrops are linked to the same rain *sound process*, and each raindrop is associated with a rain *sound process* instance. All these *sound processes* instances are activated according to the corresponding *activation profile*.

The resulting sound usually covers a large space, as determined by confirmed directivity information (see Figure 2 of Appendix A. However, only sound point is where the sound processes activation level is measured. Each sound point has an individual *sound process* instance that refers to a certain *sound process* model, e.g., a sound point on a car is assigned to a car *sound process* model, and its car *sound process* instance occurs when activated.

Activation profile :

Although the activation, as well as parameters of a *sound process*, can be all set up through a global declaration, they are better to be provided by an activation map or profile for a more interactive and sophiscated use [Schwarz et al. 2011]. In this way, *sound processes* are associated with *activation profiles* so that certain profiles activate sound processes when corresponding audio–graphic objects occur.

An *activation profile* is in fact a lookup table used to access values as a function of the positions so as to determine the activation value at each sound source/point's location. It needs to be attached to a (possibly invisible) geometry object so that it can move around in the scene to activate corresponding *sound processes* (e.g., invisible wind in trees, a moving hand in leaves, or the invisible excitation of a crowd).

We proposed to define the *activation profile* by a parametric equation, which expresses the coordinates of points of form as function of variable(s) called parameter(s). We name it *parametric profile*. Note that an alternative, not considered in our work, can be a *mesh-based profile* [Freed et al. 2010]. The *mesh-based profile* is determined by a specific mesh, in which every vertex provides an activation value.

Parametric profiles :

We determine profiles by parametric equations (functions), which are named *parametric profiles*. A graph of a function can be a line or a curve, such as Gaussian–shaped function, which may derive to 2D and 3D geometry figures by a regulated transformation, e.g., a rotation around x–axis/y–axis of a 2D shape derives a 3D form. Such an approach for profiles generation allows us to compose a great variety of complicated profiles, but only based on a basic primitives with user–settable parameters.

So we consider to propose diverse basic profile functions (1D parametric profile) which are able to derive 2D and 3D profile functions by regular revolutions. The proposed 1D parametric profile functions (depicted in Figure 4 of Appendix A) are defined by a distance to a reference point. They are linear function defined by *mindist* and *maxdist*, delta function defined by *mindist*, *maxdist*, *middle*, and *width*, and exscale function defined by *mindist*, *maxdist*, and *base* (see details in [Ding et al. 2011]. To transform them to 2D or 3D functions, it is necessary to use one of our proposed transformation methods: *revolution*, *extrusion*, or *interpolation* (see details in [Ding et al. 2011]).

5.4 Extended X3D representation

In [Ding et al. 2011], we have briefly outlined an extended X3D representation format defined in the framework of the *Topophonie* project. In the following, we explain its complete detail and discuss our motivation of the design of the extended structure and the new XML Schema design.

5.4.1 Representation of sound process

In X3D, the AudioClip and the MovieTexture are so far the only possible sound sources (the former playing samples, MP3 or MIDI files) that we have found in the

specification. As the subclasses of X3DSoundSourceNode, both elements, AudioClip and the MovieTexture, have basic parameters that control an audio data or a movie data, which can be referenced by a Sound node.

We therefore propose to extend the formalism by adding a new subclass of X3DSoundSourceNode, named TPSoundProcess, which contains the attributes corresponding to the *sound process* parameters listed in Section 5.3. Since TPSoundProcess is a node at the same level as AudioClip and MovieTexture, it inherits the properties from its super–class X3DSoundSourceNode node. TPSoundProcess is similar to AudioClip that also contains the auditive properties such as pitch, start time, stop time, and so on. Besides, TPSoundProcess supports "*model*" and "*activation*" attributes. The default attribute DEF can be used as the parameter "*identifier*".

The new class hierarchy of TPSoundProcess in X3D specification is depicted in Figure 5.1.

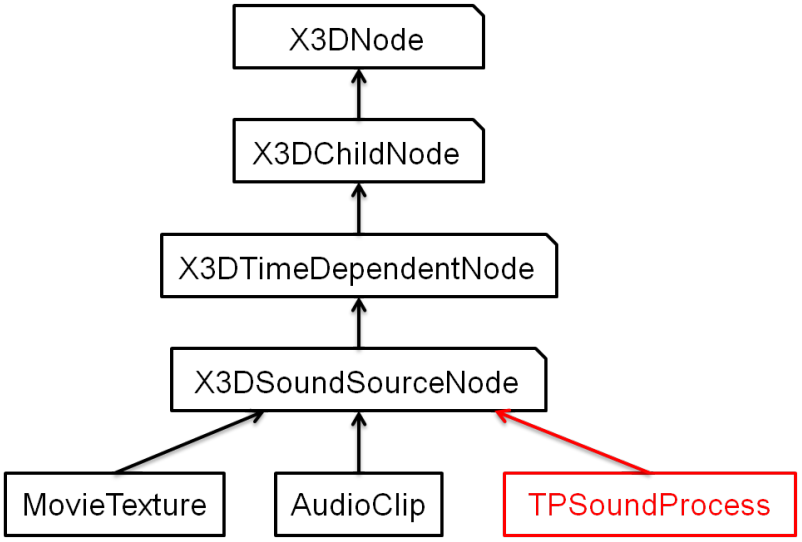


Figure 5.1: The proposed element TPSoundProcess inheriting properties from X3DSoundSourceNode.

We present the complete XML Schema designed for our modeling method in Appendix B, and an example of XML Schema for TPSoundProcess is as follows:

```
1
2 <xs:element name="TPSoundProcess">
3     <xs:annotation>
4         <xs:appinfo>
```

```
5               <xs:attribute name="otherInterfaces" type="xs:string" ↩
                    fixed="X3DUrlObject"/>
6           </xs:appinfo>
7           <xs:documentation source="http://www.topophonie.fr"/>
8       </xs:annotation>
9       <xs:complexType mixed="false">
10          <xs:complexContent mixed="false">
11              <xs:extension base="X3DSoundSourceNode">
12                  <xs:sequence>
13                          <xs:element ref="TPGeometryProfileContainer"/↩
                                >
14                  </xs:sequence>
15                  <xs:attribute name="description" type="SFString"/>
16                  <xs:attribute name="url"         type="MFString"/>
17                  <xs:attribute name="model"       type="MFString"/>
18                  <xs:attribute name="activation" type="MFFloat"/>
19              </xs:extension>
20          </xs:complexContent>
21      </xs:complexType>
22  </xs:element>
```

5.4.2 Representation of sound source

The Sound, a subclass of the abstract X3DSoundNode class, represents the placement of a sound source at a point location and its directivity and distance-based attenuation. It could be the proper node which should be used to reference to our extension of X3DSoundSourceNode, which is the *sound process*. However, Sound can be referenced to only default nodes, such as AudioClips and MovieTexture, for sound playback[2], due to the regulation limited by a content model node SoundChildContentModel. So, we need to create a new element named "TPSound" based on Sound (see Figure 5.2) in order to contain our TPSoundProcess.

5.4.3 Representation of basic profile functions in 1D

We need to introduce a new complex type TPProfileFunction which inherits the properties from x3dNode in order to represent the parametric profiles. This TPProfileFunction needs to be an "abstract" class that can derive sub–type functions in 1D,

[2]See X3D encoding documentation `http://www.web3d.org/x3d/specifications/ISO-IEC-19776-1.2-X3DEncodings-XML`

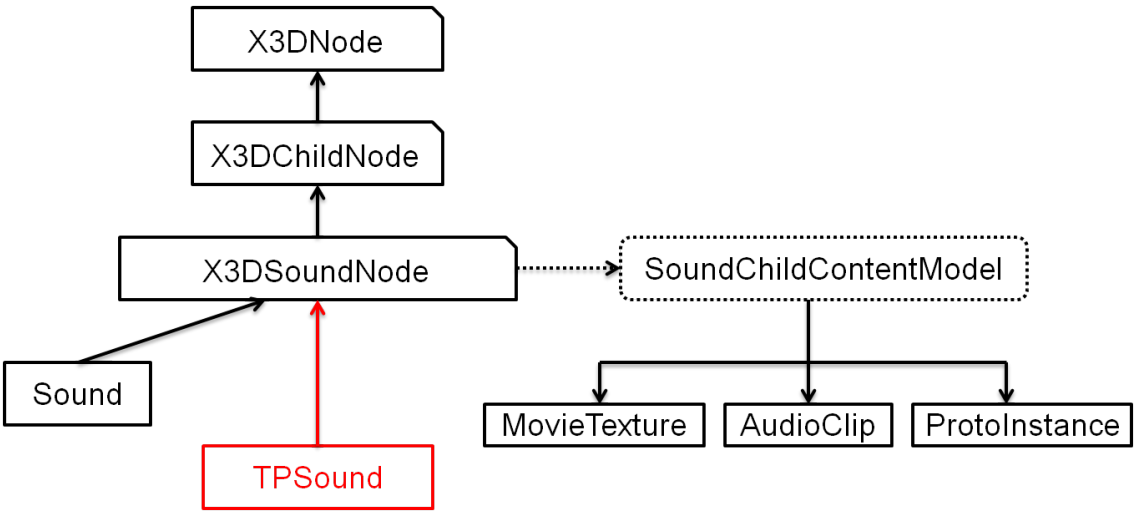


Figure 5.2: The proposed element TPSound extending X3DSoundNode.

2D and 3D, which can be defined as complex type as follows: TPProfileFunction1D, TPProfileFunction2D, addedand TPProfileFunction3D (see more details of TPProfileFunction2D and TPProfileFunction3D in Section 5.4.4). Since the functions in 2D and 3D are transformed from the functions in 1D, we defined three concrete functions (linear function, delta function, and exscale function) mentioned in Section 5.3, named TPProfileFunctionLinear, TPProfileFunctionDelta, and TPProfileFunctionExscale (see Figure 5.3).

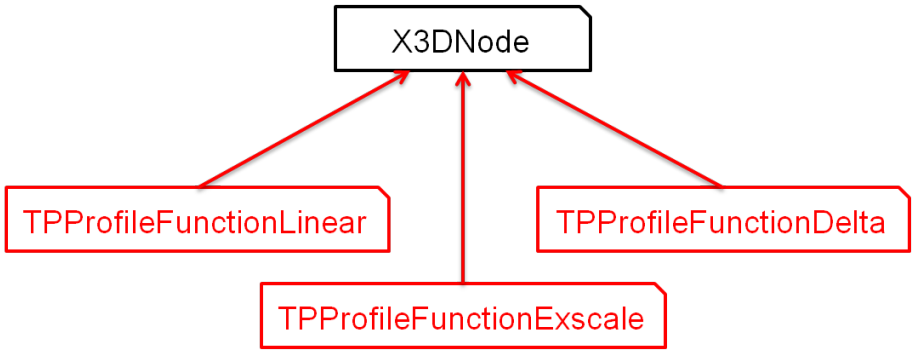


Figure 5.3: Profile function class hierarchy.

5.4.4 Representation of profile functions in 2D and 3D

According to the principles of our modeling method, one profile function needs to be associated with a geometric object (ellipse, polygon, cylinder, sphere, mesh)

that can move around in the scene. Such an object can be sometimes a complicated and self–defined geometric object, i.e., a mesh that samples the profile, a 2D or 3D color texture.

As a result, an activation profile is attached to a geometrical object which stands for a container of the profile. This container could be 2D or 3D, and can move around in our 3D scene. In our extension X3D schema (see Figure 5.4), we add a TPGeometryProfileContainer element that will contain two child nodes corresponding respectively to X3DGeometryNode and TPProfileFunction. The group ShapeChildContentModel allows the container to be any existing 2D or 3D geometry node or to create a new geometry instance by IndexedFaceSet for example.

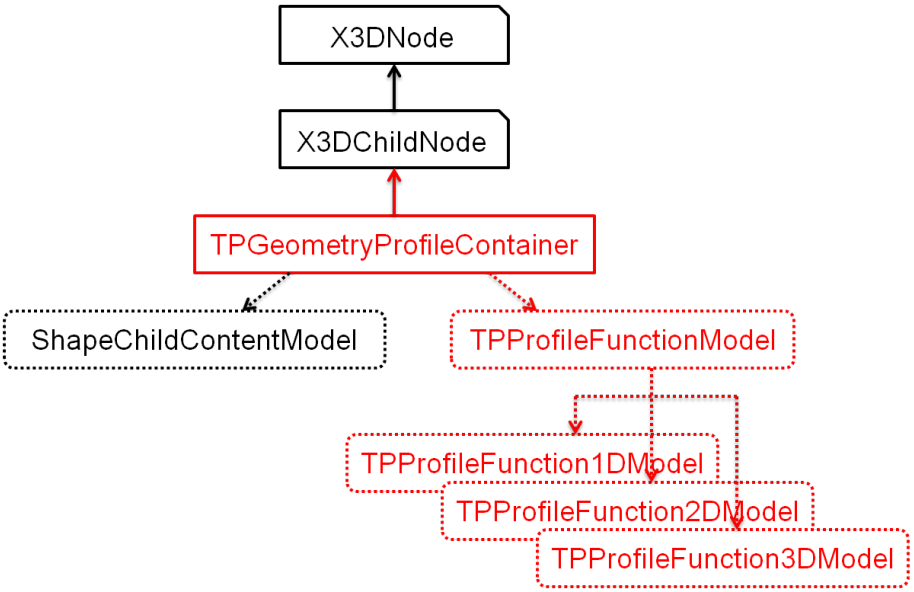


Figure 5.4: TPGeometryProfileContainer class hierarchy

As mentioned previously, due to the fixed regulation by a content model node SoundChildContentModel, Sound cannot be used to reference to a TPSoundProcess. As a result, we created TPSound. And in order to reference to TPSoundProcess, TPSound needs to be regulated by a new content model node, named TPProfileFunctionModel. In other words, the TPProfileFunctionModel in TPGeometryProfileContainer works as the SoundChildContentModel in Sound, so that TPProfileFunctionModel contains all the models of profile functions which are classified by dimension (TPProfileFunctionModel1D, TPProfileFunctionModel2D, TPProfileFunctionModel3D). Every model contains different profile functions (for example, TPProfileFunctionModel1D contains the TPProfileFunction1D type of elements, such as TPProfileFunctionLinear, TPProfileFunctionDelta, and

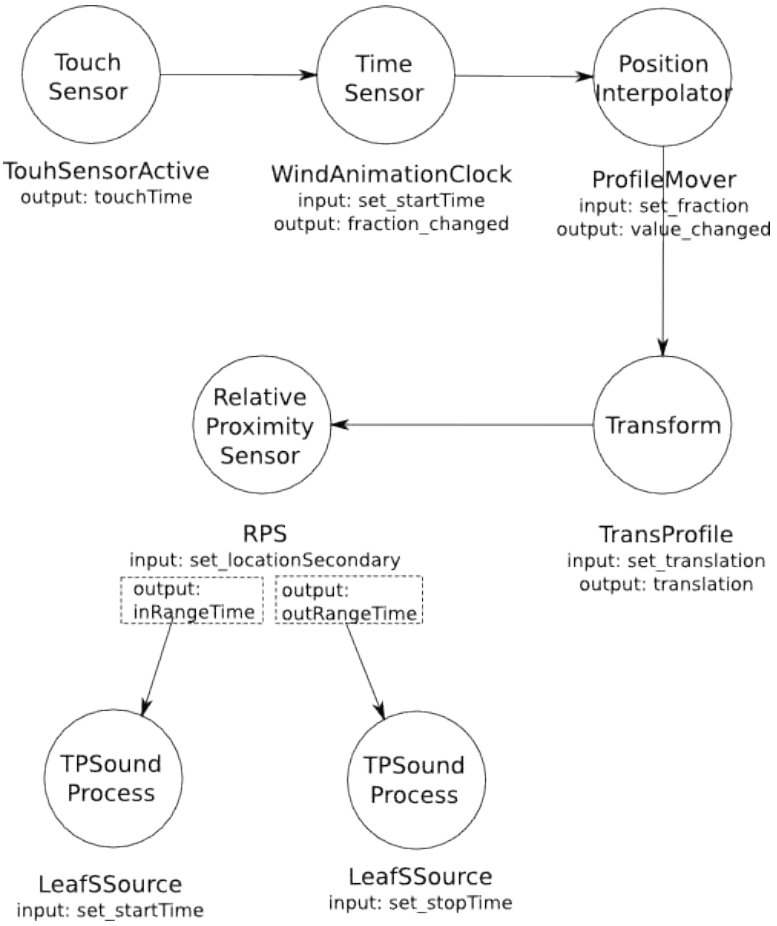


Figure 5.5: An example of event chain for an audio–graphic scene

TPProfileFunctionExscale).

5.5 X3D audio–graphic modeling

We have seen all the concepts and definitions about our modeling method, as well as the extended standard of X3D for adapting such ideas in the the above discussions. Now we present a guideline of how to realize the general audio–graphic modeling in X3D through an event chain design of the "tree in the wind" example.

In X3D, the audio–graphic scene is generated with an event chain linked by the connection node ROUTE. We introduce here one general event chain (see Figure 5.5) that simulates a wind profile moving in the X3D scene to activate the sound process when wind profile meets the sound sources (leaves).

The procedure of the event chain is explained as follows:

1. First of all, the X3D scene will have a sensor to activate the movement of wind profile. The TouchSensor is one example of sensor, and the TouchSensor is activated when a pre-defined area is clicked, and the activation time is recorded.

2. The TouchSensor outputs the activation time (mentioned above) to next sensor which is TimeSensor.

3. The TimeSensor then outputs the changed fraction (according to the passing time until activation time of TouchSensor) to the PositionInterpolator.

4. The PositionInterpolator begins to change its value according to its interpolator pre-definition. And the changing value will be sent to the transformation of the wind profile, then the wind profile will move according to the PositionInterpolator.

5. We declare an external prototype named RelativeProximitySensor to simulate the detection of the collision between objects. The computation of this simulation prototype is defined in another X3D file. Every sound source should be associated with a RelativeProximitySensor instance. RelativeProximitySensor detects whether the sound source enters in the wind profile, while the according sound process will be activated when the sound source enters in the wind profile.

5.6 Discussion and conclusion

In this chapter, we have presented the principles of a new modeling method of audio–graphic scenes based on the concept of *sound process*. Moreover, we have extended the standard 3D file format, X3D, to represent audio–graphics scenes based on our modeling method, since the traditional sound elements in X3D is not refined enough for a complex information-exchange between the audio and graphical processes. We also provide a general guideline for the application of our audio–graphic modeling based on *sound process*.

Our contribution of this work is that we generate new knowledge and further understanding of audio–graphic modeling framework through an XML based formalism. Because of the complication of auditory event under different conditions,

we cannot realize all situations or scenarios. Nevertheless, thanks to the extensionality of X3D, the *Topophonie* extended X3D schema can be further extended in the future to take in charge additional constraints, which, we believe, will include the audio–graphic LOD selection features.

Part IV

Evaluation of crossmodal perception on multi–LOD system

Chapter 6

Design and evaluation of audio–visual LOD system

In this chapter, we focus on the investigation of the role of crossmodal LOD audio–graphic scenes. Our work attempts to discover the crossmodal perception phenomena and interactions based on multi–modal LOD. We begin with the question of how to jointly represent the scene objects at different LODs in both visual and audio modalities for doing so.

We have tried two approaches for rendering GSLODs. The first approach is based on an impostor method for both GLODs and SLODs, while in the second approach, we use mesh simplification techniques for generating GLODs. We chose these two approaches so as to cover the most common methods of GLODs used in computer graphics. To evaluate the impact of GSLODs on the first approach of rendering, we used psychophysical methods for designing perceptual experiments. In the following sections, we will present the GSLOD rendering based on an impostor method and also the perceptual evaluation of GSLODs.

6.1 Approach

In our work, we are interested to investigate how the scene objects can be jointly represented at different LODs in both visual and audio modalities. In our recent work [Ding et al. 2011], we define a graphic object that makes or has sound as an *audio–graphic object*. Note that the location of an audio–graphic object usually coincides with the sound source location. Here, our approach for multimodal LOD is simple: we represent the graphic objects through GLOD generation, and repre-

sent the *audio–graphic objects* (originated from the graphic objects) through SLOD generation. GLODs and SLODs are pre–generated respectively by using k–means clustering algorithm. Since GLODs and SLODs are designed to be paired at each LOD, we call a GLOD–SLOD pair GSLOD. According to the distance between the observer and the objects, the appropriate GSLOD is selected for audio–graphics rendering.

We developed a realistic audio–graphic tree scene with wind-based motion and sound generation. Two separate but dependent engines were determined for rendering graphics and sound: *Graphics Engine* for graphics rendering supporting GLOD selection, and *Audio Engine* for sound rendering and SLOD selection. A preprocessing stage is conducted respectively for GLOD and SLOD generation by k–means clustering algorithm as mentioned previously. Figure 6.1 shows the schema and functionality of the audio–graphics engine and preprocessor. In this figure, IAE is an Interactive Audio Engine that was developed by IRCAM partner for corpus-based sound synthesis and SLOD generation, while Woody3D[1] is an open source real–time animated 3D tree modeling and real–time rendering tool which provides an API (Application Programming Interface) functionality for permitting programmable access to the tree geometry. Based on Woody3D, we developed a *Graphics Engine* which accomplishes the GLOD selection and rendering. The *Graphics Engine* has another important task which is to calculate the sound parameters based on geometry and motion data. By receiving the OSC (Open Sound Control) messages containing sound parameters that are calculated in the *Graphics Engine*, the *Audio Engine* performs the SLOD selection and rendering.

6.1.1 Corpus-based sound synthesis

IAE engine generates an up–to–date sound and SLOD based on corpus–based sound synthesis, which has been briefly mentioned previously in Section 1.3.1. Such a novel sound synthesis technique is applied in our implementation for audio rendering, since it is appropriate for rendering granular audio scene such as tree(s).

Corpus-based concatenative sound synthesis (CBCS) [Schwarz 2007] is a recent method for offline or real-time procedural generation of audio, used in sound design, music composition and performance, and also interactive multimedia applications such as environmental sound texture synthesis [Schwarz and Schnell 2010]. It can be

[1] http://www.woody3d.com/

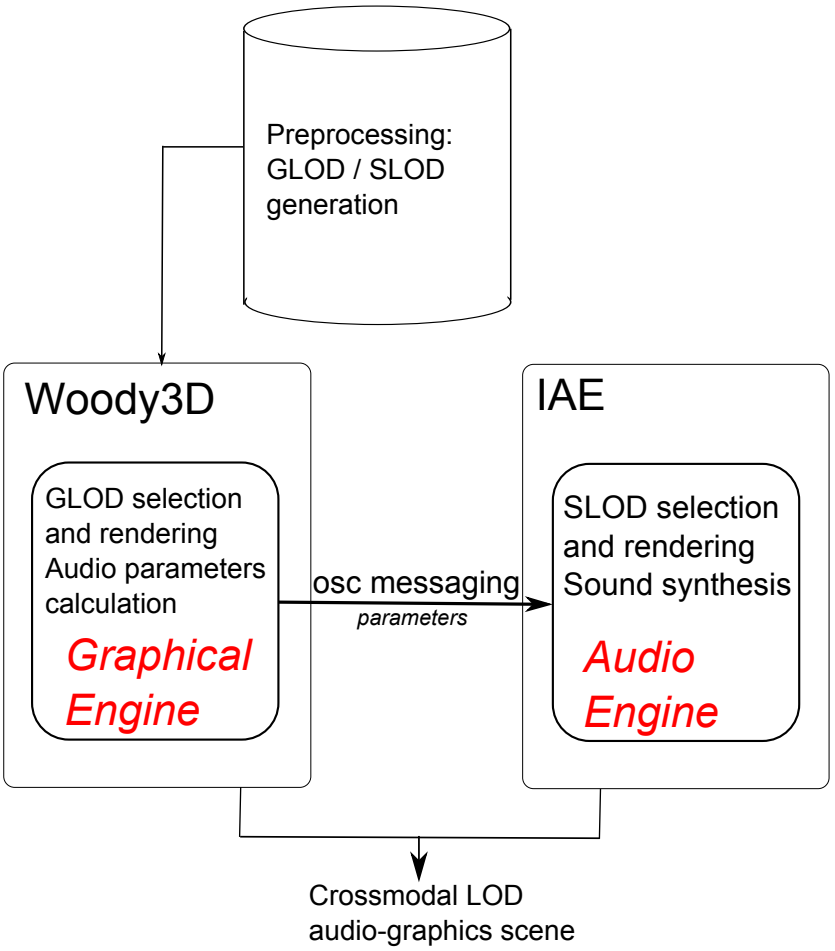


Figure 6.1: Audio–Graphics Engine and GSLOD Preprocessor

seen as a content-based extension of granular synthesis according to audio descriptor analysis.

Corpus-based sound synthesis systems build up a database of pre–recorded or live-recorded sound by segmenting them into *units*, usually of the size of a note, grain, phoneme, or beat, and by analysing them for a number of sound descriptors, which describe their sonic characteristics. These descriptors are typically pitch, loudness, brilliance, noisiness, roughness, spectral shape, etc., or meta-data, such as sound source class, labels, etc., that are attributed to the units. These sound units are then stored in a database (the *corpus*). For synthesis, units are selected from the database that are closest to given *target* values for some of the descriptors, usually in the sense of a weighted Euclidean distance.

The selected units are then concatenated and played, after some possible sound

transformations such as pitch and amplitude change, filtering, and so on.

6.1.2 Sound process modeling

When audio–graphic objects occur in a scene, visual and audio modalities are synchronized in time and space such that the audio–graphics objects share a common process for both modalities. In an audio–graphic scene, the graphics information is often visualized to viewer, while the audio information is relatively invisible and perceived only when activated. Different from work on acoustics [Tsingos et al. 2001] or sound synthesis [O'Brien et al. 2002], our work is focused on how to place the sound sources in a 3D scene.

In the example of sound made by leaves in the wind, we assume that every leaf is a sound point which is linked to a sound process, and the wind can be depicted as a kind of profile. When the wind profile moves through the leaves, the sound processes will be activated.

We treat our tree *Audio Engine* as a collection of independent *sound processes*, that are controlled by *activation profiles*. In the tree scene example, one sound process takes care of the audio rendering of the whole tree, and the activation level corresponds to the strength of the wind. The activation level also activates the movement of branches in a combined audio–graphic process [Schwarz et al. 2011].

6.1.3 Impostor/Image–based GLOD

The GLOD for tree is defined in a preprocess before rendering. To coherently integrate with the idea of SLOD [Schwarz et al. 2011], of which we will see more details in Section 6.1.4, we established three GLODs in our case:

GLOD1 foreground: the most detailed presentation of granular objects. They should be the original geometry and sound of the audio–graphic objects. Every granular object is presented as the smallest unit. In our tree example to be introduced below, for instance, one granular object is a textured quadrilateral that contains the smallest group of leaves.

GLOD2 middle ground: the less detailed presentation of object groups. They are presented based on the granular objects in GLOD1. A clustering method can

be applied for grouping objects, for example, and every resulting cluster can be placed by a new coarser primitive/object.

GLOD3 background: the coarse object presentation, which may be obtained by the same grouping process as GLOD2 but with a smaller number of resulting clusters.

In practice, we made a realistic 3D tree animation based on a commercial tree engine Woody3D. We applied the k–means clustering algorithm on quadrilaterals for grouping them so as to generate GLOD2 and GLOD3. Naturally, leaves are considered as audio–graphic objects, then the GLOD1, which is the full–detail rendering, here should be the textured quadrilaterals of the smallest unit of leave groups, as mentioned above. We chose the k–means clustering algorithm to group the quadrilaterals, because it is simple and efficient for data partition in space. And it allows us to easily control the desired number of quadrilaterals for a simplified representation. With the numbers of clusters (re–grouped leave quadrilaterals) freely chosen for GLOD2 and GLOD3, the *Graphics Engine* pre-performs two times the k–means clustering algorithm on granular objects in order to have those three levels of detail for graphics. For example, based on approximately 1120 quadrilaterals for GLOD1, we chose 700 quadrilaterals and 350 quadrilaterals as numbers of clusters for k–means clustering algorithm for generating GLOD2 and GLOD3 (see Figure 6.2).

Typically, our GLOD preprocessing is known as the discrete LOD approach [Luebke et al. 2002], so at run time the more distant the object, the coarser its LOD. This discrete LOD framework is often applied in real-time 3D applications such as video game, and in our work such GLODs can be simply combined with SLODs which also use the distance as LOD selection factor. We present the technique for SLOD rendering in Section 6.1.4.

In Figure 6.3, the three screenshots, from top to bottom, show the tree geometry without (on the left) and with (on the right) texturing in GLOD1 (on the top), GLOD2 (in the middle), and GLOD3 (on the bottom), respectively. Note that the branches are not relevant for audio–graphic objects, so they are in full detail rendering whatever the selected LOD is. Specifically, we have 1120 textured quadrilaterals in GLOD1 for the tree, and by applying two times the k–means clustering with desired input cluster numbers, we obtain 700 and 350 new quadrilaterals for GLOD2 and GLOD3, respectively.

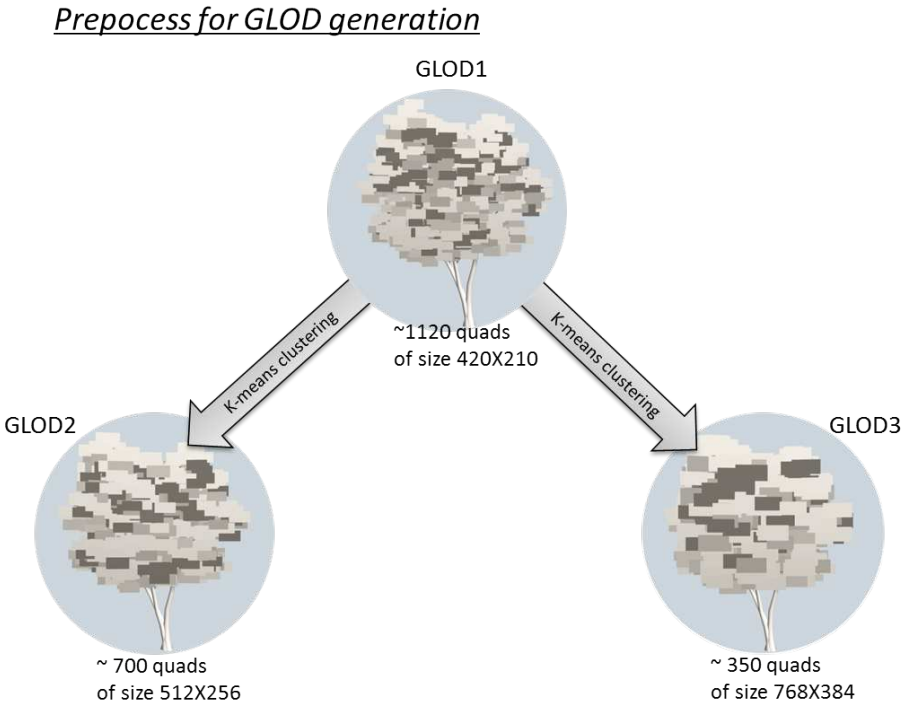
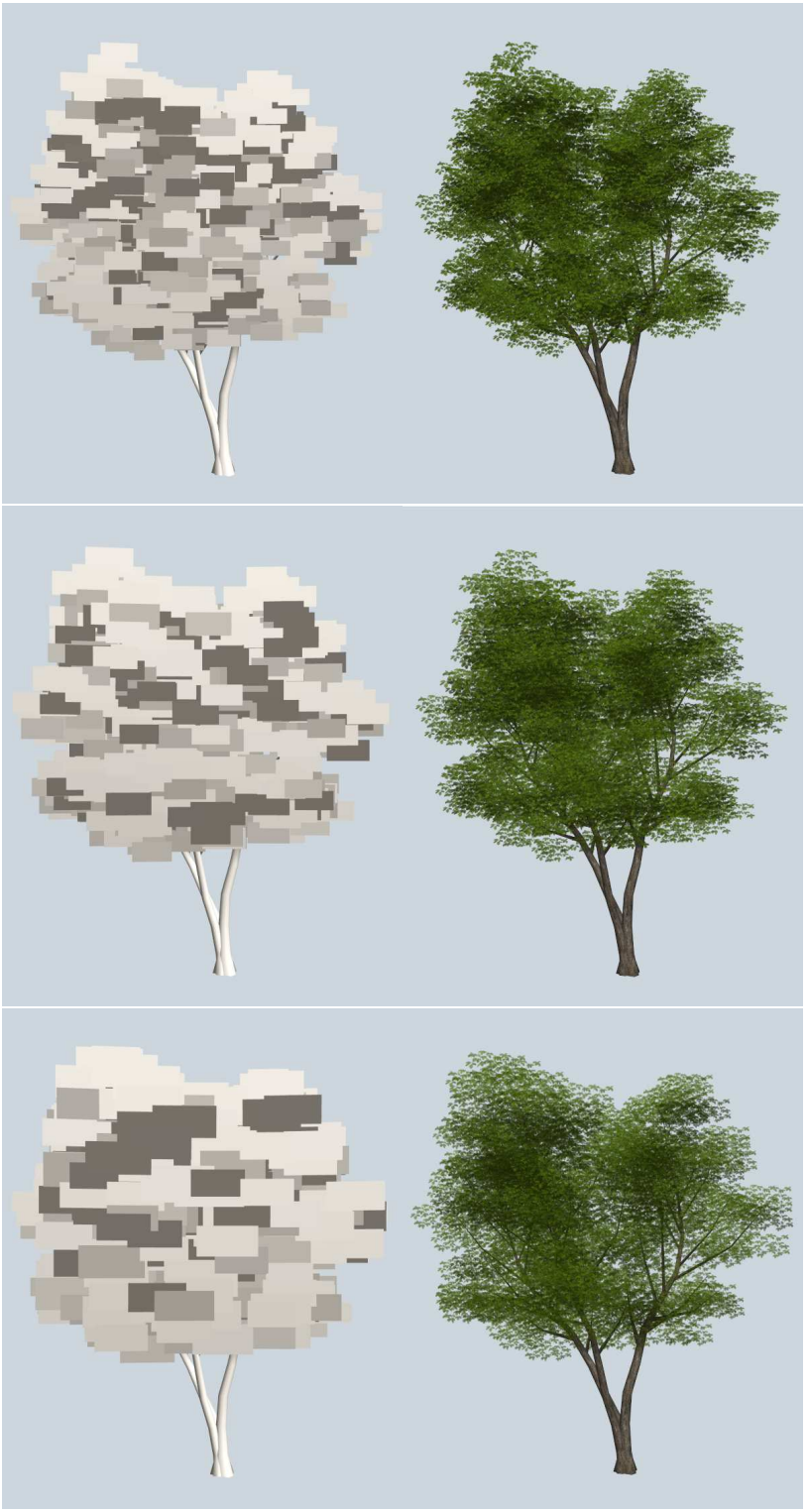


Figure 6.2: GLODs generation by k–means clustering

The new quadrilaterals used to replace the grouped quadrilaterals are naturally larger than the initial ones. As a result, we chose three different sizes of quadrilaterals for GLOD1 to GLOD3, which are 420×210, 512×256, and 768×384. The textures used for different GLODs are also different since the higher the LOD the more leaves are represented in the texture. The texture of dimension 420×210 used for GLOD1 contains forty leaves in the image, which makes $1120 \times 40 = 44800$ leaves in total for one quadrilateral. The two other textures of dimension 512×256 and 768×384 for GLOD2 and GLOD3 represent respectively 64 and 128 leaves (see Figure 6.4). Consequently, we have almost the same number of leaves represented at every GLOD. The tree may have few changes in foliage intensity at different GLODs even though we have carefully kept the same number of leaves at each LOD and made fine regrouping result. And in Figure 6.3, we can perceive that few visible changes in foliage intensity of our GLODs have occurred from our GLODs generation.

The animation of leaves and branches is realized through a simulated wind producer mechanism in the *Graphical Engine*, which calculates the wind intensity lev-

Figure 6.3: The tree scene in three GLODs.



Tree geometry without (left) and with (right) texturing represented in GLOD1 (top), GLOD2 (middle) and GLOD3 (bottom).

Figure 6.4: The textures for three GLOD.

From left to right, the three textures are used on the quadrilaterals for GLOD1, and GLOD2 and GLOD3, respectively.

els and wind direction for every textured quadrilaterals and branches according to time and wind dynamics. In each frame, the wind intensity values and wind direction are updated by a parameterized randomizer using a time value. The wind intensity values range from 0.0 to 1.0 (corresponding to 0 to 100 mph or 0 to 160 km/h). According to these real time wind intensity values and wind directions, the leaves/quadrilaterals are updated in real time with wind–caused movement. For graphics rendering, the wind intensity and wind direction define the leave motion; and for audio rendering, all wind intensity values at every quadrilateral may be considered as a sound *activation profile* so that it defines exactly the activation of sound. As a result, such a sound activation profile generalized by wind intensity values is geometrically a point cloud of tree leaves/quadrilaterals. Each intensity value and its relevant quadrilateral's coordinates are sent to the *Audio Engine* in order to generate the real time SLOD corpus–based sound.

6.1.4 Impostor SLOD and integration

To our knowledge, the study of LOD for sound rendering is not as popular in the sound synthesis research community as that in computer graphics. In our recent work [Schwarz et al. 2011], we introduce a new technique for SLOD rendering. The SLODs are selected according to the distance factor below:

SLOD1 foreground: individually driven sound events and sound behaviors. When we are very close to an audiographic cluster, for example consider rain drops on tree leaves, each drop collision should be heard and seen individually.

SLOD2 middle ground: group-driven sound event, statistical behaviors. Above a certain density of events, when they can hardly be isolated any more, they

play stochastically, according to a sound behavior preset. This limit is passed when sources are farther than a certain distance from the listener.

SLOD3 background: sound impostors. Even further away, sources can be simply rendered by continuous audio impostors such as audio files, or take advantage of the scene depth partition or spatial clustering knowledge to dynamically mix groups of procedural impostors according to the view point and the evolution of the scenario.

The distance between the sound source and the listener is used as the selection factor for SLOD. Since in the tree audio–graphical scenes the listener and the viewer are the same, the distance becomes the common selection factor for both GLOD and SLOD. In practice, we chose to make GLOD and SLOD paired matching. A simple connection of the three levels for SLOD and three graphic levels of detail should be very appropriate. Consequently, the SLOD foreground is associated with the GLOD1; the SLOD middle ground is associated with the GLOD2; and the SLOD background is associated with the GLOD3.

In most cases, sound objects and graphics objects are related in time and in space, so called *audio–graphic objects*, as a sound source is always situated in a graphic object or a group of them. Therefore, we consider that in an audio–graphics system, either the audio engine controls the graphical engine or the other way around. Concretely, when *graphical engine* controls, it is not only used for GLOD generation and graphics rendering but also used to calculate the information sending to *audio engine* for generating SLODs sound. When *audio engine* controls, it generates SLOD sound and calculates information for *graphical engine* to generate GLODs. In our work, the *Graphical Engine* has control over the GSLOD selection. Specifically, it performs both GLOD rendering and GLOD&SLOD selection, as well as indispensable parameterization for SLOD rendering in *Audio Engine*. It sends data such as cluster (sound source) sizes, cluster centroids and wind intensity (see Section 6.1.3) to the *Audio Engine* which can then produce real–time SLOD sound.

6.1.4.1 Implementation of the audio engine and the corpus

The *Audio Engine* used in our work is a granular and corpus-based engine IAE, developed as a cross-platform C++ library, integrated in Max/MSP[2], Unity3D and

[2]Max/MSP is a musical software that allows programmable sound synthesis. `http://cycling74.com/products/max/`

iOS. IAE realizes corpus-based sound synthesis as mentioned in Section 6.1.1 and generates SLOD sounds with the data sent by the *Graphical Engine*.

The sound corpus is constituted from recordings of various types and sizes of tree branches, realized in a recording studio by agitating and manipulating the branches in multiple ways. These raw recordings are then combined into a sound corpus and automatically segmented into equal-sized units of 500 ms without any manual post-processing except de-rushing and trimming. The size of the units has an effect that the fine temporal structure of the leaves' rustling is kept in contact.

Since we have used a kind of audio descriptors, the corpus can be accessed by wind intensity *loudness* descriptor which allows to specifically play the segments corresponding to a certain agitation of the branches, convincingly representing the rustling of the leaves at a certain wind intensity.

Moreover, because of the richness of the corpus, the granular re–synthesis generates a varied, never repeating sound texture, that is nevertheless precisely controllable in its sound character.

Figure 6.5 shows the interface of IAE that allows to load raw recordings, and generates on demand corpus sounds according to SLOD and tree location with reference to observer. It is an developable interface integrated in Max/MSP.

6.1.4.2 Sound impostors

As one tree branch roughly corresponds to what is represented in the GLOD1 texture, one audio source played from this corpus represents SLOD1. In order to generate sound impostors [Schwarz et al. 2011] for SLOD2 that are not just fixed samples but remain dynamically controllable, we record 10 seconds of a mix of four SLOD1 audio sources, at 10 different levels of activation, equally spaced between the minimum and maximum wind intensity.

These 10 recordings then form the source material for an SLOD2 corpus that is accessed by activation only, but the granular playback is parameterized with a large random variation of the playback position, thus generating a never-repeating sound texture for each activation level that encompasses the sound of a group of four tree branches.

For SLOD3, the same procedure is used, recording 10 seconds long sound impostors generated from four audio sources of SLOD2, for 10 levels of activation. The

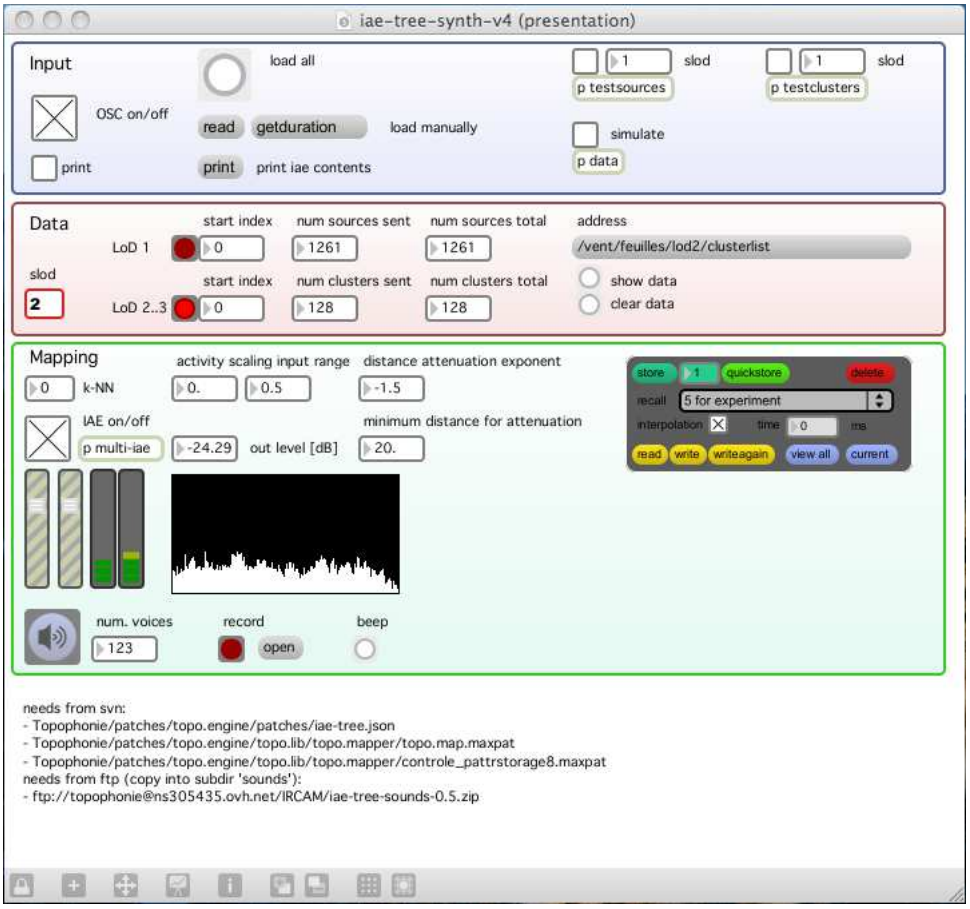
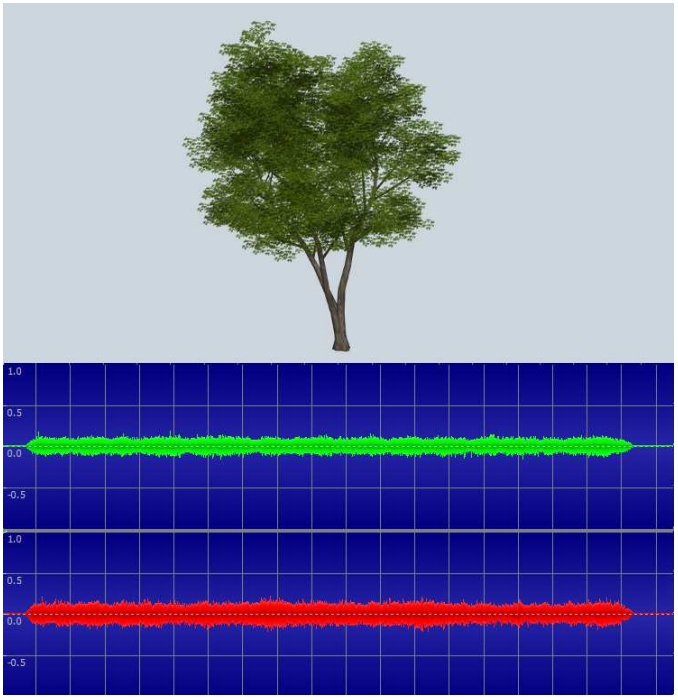


Figure 6.5: IAE developable interface integrated in Max/MSP.

Figure 6.6: A tree rendered in GSLOD1.

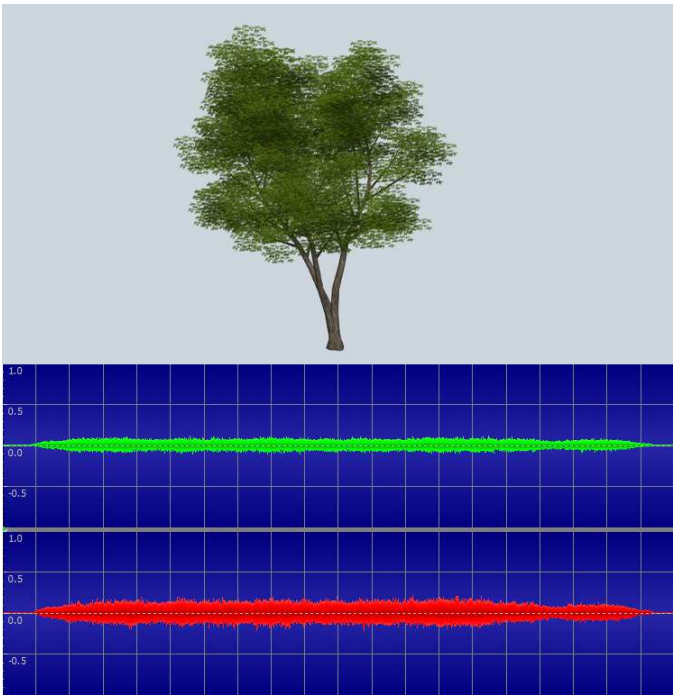


On the top is the tree in image, and on the bottom is the tree sound waveform.

waveforms of audio sample in each SLOD are showed in Figures 6.6, 6.7, and 6.8.

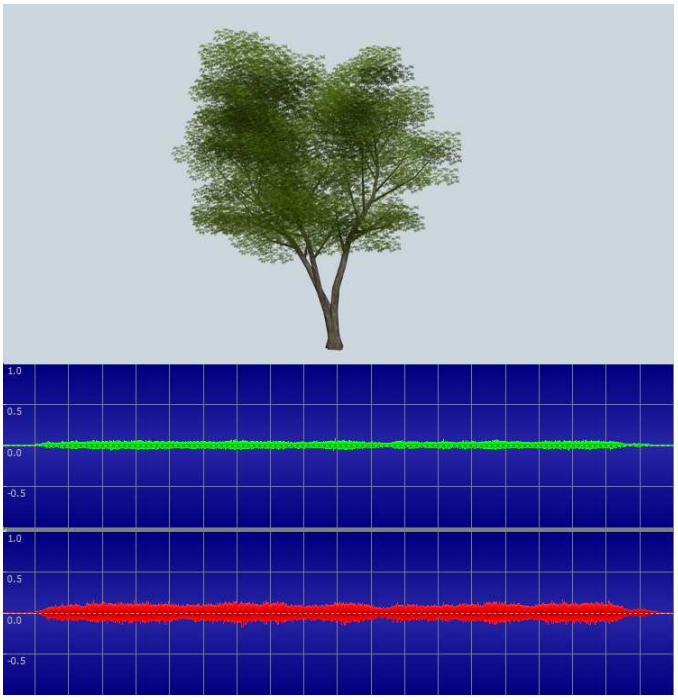
As the tree graphics contains about 44800 leaves in total, we have considered that 512 sound sources were selected for SLOD1, 128 sources for SLOD2, and 32 sources for SLOD3 so as to maintain an approximately equal number of leaves in audio simulation as well as in graphics. The configuration of GSLOD is given in Table 6.1. The preprocessing also pre-performed three times k-means clustering by clustering the granular objects (1120 quadrilaterals) into 512, 128 and 32 clusters based on the information of tree geometry. Besides, by calculating real–time wind intensity for each cluster, the *Graphics Engine* sends the appropriate real–time parameters to *Audio Engine* for generating SLOD sound, as mentioned previously. Such a crossmodal pairing of GLOD and SLOD, is thus called GSLOD. We show three GLODs and their paired three SLODs in Figures 6.6, 6.7, and 6.8, respectively.

Figure 6.7: A tree rendered in GSLOD2.



On the top is the tree in image, and on the bottom is the tree sound waveform.

Figure 6.8: A tree rendered in GSLOD3.



On the top is the tree in image, and on the bottom is the tree sound waveform.

GLOD k–means clustering algorithm used to build the granular objects (leave quadrilaterals).	**SLOD** k–means clustering algorithm used to build the audio–graphics objects (sound sources).
GLOD1: o 1120 quadrilaterals o Texture: $\sim$ 40 leaves o Total: $\approx$ 44920	SLOD1: o 512 sound sources o Source: 1 impostor o Impostor: 80 $\sim$ 100 leaves o Total: 40960 $\sim$ 51200
GLOD2: o 700 quadrilaterals o Texture: $\sim$ 64 leaves o Total: $\approx$ 44800	SLOD2: o 128 sound sources o Source: 4 impostors o Impostor: 80 $\sim$ 100 leaves o Total: 40960 $\sim$ 51200
GLOD3: o 350 quadrilaterals o Texture: $\sim$ 128 leaves o Total: $\approx$ 44800	SLOD3: o 32 sound sources o Source: 16 impostors o Impostor: 80 $\sim$ 100 leaves o Total: 40960 $\sim$ 51200

Table 6.1: Configuration of GLOD and SLOD

6.2 Experiment

Based on Approach I (see Section 6.1), we have performed a perceptual experiment by using a method of limits which is to find the discrimination thresholds, in order to explore the role of crossmodal LOD in audio–graphic scene for perceptual discrimination. Inspired from the experimental design of LOD in 3D tele-immersive video [Wu et al. 2011], we used in our experiment the ascending approach for method of limits, which is one of the traditional experimental methods [Gescheider 1997] to investigate the limit of perception (i.e., discrimination thresholds) for different LOD conditions depending on purpose. Such an experimental method is well suited for audio stimuli and graphic stimuli respectively, and we believe that it should be also well suited for audio–graphic stimuli, since the audio and graphic events are synchronized both in space and time.

In experiments, participants were asked to judge whether the test stimuli (GLOD2 / GLOD3 / SLOD2 / SLOD3 / GSLOD2 / GSLOD3) are different from corresponding reference stimuli (GLOD1 / SLOD1 / GSLOD1), respectively. A succession of test stimuli with increasingly varied intensity are judged by participants in order to assess the Just Noticeable Difference (JND) of stimulus intensity. Empirically, one stimulus intensity is supposed to be confirmed as a threshold when it is judged to be "different" from a reference at a probability of $p = 50\%$ across different users and different trials.

6.2.1 Method of limits

The method of limits is one of the three classical psychophysical methods for testing participants' perception of stimulus detection and discrimination. Note that the other two are the method of constant stimuli and the method of adjustment. In psychophysics, the method of limits is usually used in perceptual experiments to evaluate a detection or discrimination threshold of the experimental stimuli, such as pure tones varying in intensity or lights varying in luminance. Typically, a detection/discrimination threshold is considered to be the point at which a test stimulus can be judged to be different from the reference stimulus above a given frequency. The often used probability p is 50% [Gescheider 1997]. The participants are presented the appropriate stimuli from lowest to highest intensity (ascending method) or reverse (descending method). After each trial (reference stimulus and test stim-

ulus), a participant answers whether he/she perceives the difference, and one series of trials ends once he/she perceives the difference or when all the stimuli are shown. The procedure of an ascending method for one series is described below:

1. A series begins with a stimulus intensity well below threshold (by conjecture).
2. Stimulus intensity is increased using small steps until it reaches a given limit.
3. In each trial the participant responds whether he/she can perceive the stimulus or difference between stimuli.
4. An individual threshold for this series/participant is estimated at the moment the participant finds/perceives the stimuli, and then this series ends.

After all series of trials are done, the final detection/discrimination threshold should be the one with the probability equal to 50%.

6.2.2 Participants and apparatus

We have invited 26 people with normal or corrected vision and normal hearing, aging from 20 to 35, to participate our experiment. During the experiment, participants were provided with a Dell flat panel LCD monitor display with resolution of 1600×1200 pixels and a Sennheiser HD headphone. In a sound-proofing room, participants sat 0.5 meter away from the monitor and were equipped with headphones. The graphics is programmed in OpenGL and C++ with 45 degrees as fovy, $\frac{4}{3}$ as aspect and 0.1/1000 as zNear/zFar. The window size for presenting graphics is 1600×1200.

6.2.3 Stimuli

Empirically, when an object is farther, it is more difficult to perceive the detail, which is one of the main principles of LOD. We used the ascending method of limits by reducing the distance (between observer and observed objects) from far enough to near enough in order to get a succession of ascending stimulus intensity. However, distance step is difficult to estimate. We decided to choose a small enough step–size to have sufficient stimulus intensities in image–space defined distance.

To ensure the beginning stimulus intensity is well below the threshold, we started from a far enough distance so that the pixel height of tree is about 30.0. In the next

stimulus, we increased by 30% the pixel height. We used the same step size on the continuous stimuli to make sure that the tree pixel height in current stimulus is 30% smaller than the one in next stimulus (see Eqn. (6.1), where n indicates the n–th stimulus) until the distance is near enough to have 908.6 as pixel height of tree. Consequently, we have 14 distances that make 14 pairs of stimulus (reference stimulus vs. test stimulus) for each condition.

$$Pixel\_height_n = 30.0 \times (1 + 30\%)^{n-1} \tag{6.1}$$

We also included identical stimuli, i.e., reference stimulus paired with itself. The purpose is to check whether participants discriminate identical stimuli.

6.2.4 Procedures

Similar to the experiment performed in [Wu et al. 2011], our approach is to ask each participant whether he/she finds the test stimulus different from the reference stimulus. The experiment follows the ITU-R BT.500 standard [ITU 2009]. Every stimulus, which is a section of tree animation in mono-modality (audio/visual modality) or in multi-modalities (audiovisual modalities), lasted 8 seconds. Between a reference stimulus and corresponding test stimulus, there is a pause of 2 seconds (white and mute scene). 26 people with normal or corrected vision and normal hearing, aging from 20 to 35, have participated our experiment. They answered the question after each pair of stimuli (reference and test stimulus).

We consider that the distance is the factor that determines the limits of perception for GLOD, SLOD and GSLOD (see Section 6.2.3). Therefore, the experiment contains six test conditions, GLOD2, GLOD3, SLOD2, SLOD3, GSLOD2 and GSLOD3. And in each condition, the stimulus intensity is varied by reducing the distance between the observer and the tree object. The corresponding reference stimuli used to compare with test stimuli are GLOD1, SLOD1, and GSLOD1. We also have mixed up three conditions of identical stimuli, which are conditions *GLOD1 vs. GLOD1*, *SLOD1 vs. SLOD1*, and *GSLOD1 vs. GSLOD1* (see Table 6.2). In practice, every participant would test all these 9 conditions, and the 9 conditions were never showed in the same sequence to each participant.

In Table 6.2, the conditions *GLOD1 vs. GLOD2* and *GLOD1 vs. GLOD3* were the tests of visual modality in GLOD. In these experimental conditions, participants

Condition 1	*GLOD1 vs. GLOD2*
Condition 2	*GLOD1 vs. GLOD3*
Condition 3	*SLOD1 vs. SLOD2*
Condition 4	*SLOD1 vs. SLOD3*
Condition 5	*GSLOD1 vs. GSLOD2*
Condition 6	*GSLOD1 vs. GSLOD3*
Condition 7	*GLOD1 vs. GLOD1*
Condition 8	*SLOD1 vs. SLOD1*
Condition 9	*GSLOD1 vs. GSLOD1*

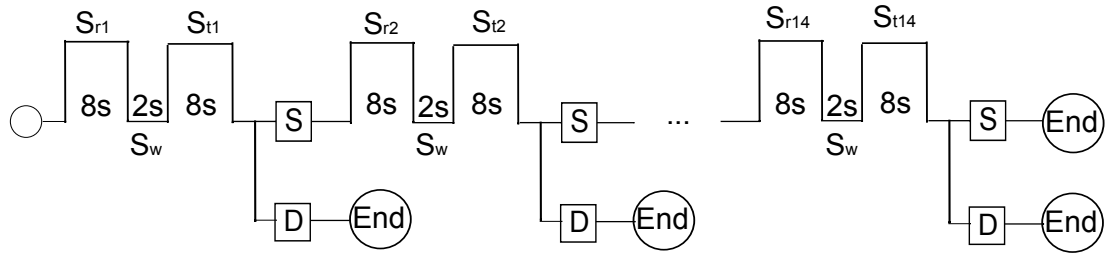
Table 6.2: Conditions

only see the graphics and there is no sound present in the scene. The reference stimuli are always in GLOD1 at appropriate distance, and test stimuli are in GLOD2 and GLOD3, respectively. Participants answer whether they perceive difference after observing each pair of stimuli (reference stimulus and test stimulus). The conditions *SLOD1 vs. SLOD2* and *SLOD1 vs. SLOD3* are the tests of audio modality in SLOD. In these parts of experiment, participants hear sound but do not see graphics. Similar to visual modality conditions, the reference stimuli are in SLOD1, while the test stimuli are in SLOD2 and SLOD3, respectively. And the conditions *GSLOD1 vs. GSLOD2* and *GSLOD1 vs. GSLOD3* are the tests of audiovisual modalities in crossmodal LOD. The reference stimuli are in crossmodal LOD1, so called GSLOD1, and the test stimuli are in GSLOD2 and GSLOD3, respectively. Note that *GLOD1 vs GLOD1*, *SLOD1 vs. SLOD1*, and *GSLOD1 vs. GSLOD1* are the three conditions of identical stimuli.

As mentioned previously, we applied an experimental procedure which is referred to the ascending method [Gescheider 1997]: the procedure ends up when the participant finds the difference between the reference stimulus and the test stimulus, and the stimulus intensity where the discrimination happens is then recorded. The procedure can also end up anyway after the last pair of stimuli, if one participant cannot find difference during one whole procedure. In practice, for each condition, 14 pairs of stimulus are supposed to be presented to participants from far to near (see Figure 6.9).

Table 6.3 is a sample form for logging the answers of a participant. The first row in Table 6.3 indicates the labels signifying 14 pairs of stimulus. And the first column indicates the conditions. The remaining cells are used to record the answers. D means “different”; S means “similar/same”. The sign “–” means no answer, when

Figure 6.9: The procedure of experiment for the Approach I



S_{r1}/S_{r14} signifies the reference stimulus at the first/14th distance; S_{t1}/S_{t14} signifies the test stimulus at the first/14th distance; S_w signifies the white scene or mute scene; D/S indicates different/same for the answer of participant; End means the condition ends here; 8s/2s indicates the scene lasts 8/2 seconds.

the procedure ends in advance by an answer "different".

	1	2	3	4	5	6	7	8	...	13	14
GLOD1 vs. GLOD2	S	S	S	S	D	–	–	–	...	–	–
GLOD1 vs. GLOD3	S	S	D	–	–	–	–	–	...	–	–
SLOD1 vs. SLOD2	S	S	S	S	D	–	–	–	...	–	–
SLOD1 vs. SLOD3	S	S	D	–	–	–	–	–	...	–	–
GSLOD1 vs. GSLOD2	S	S	S	S	D	–	–	–	...	–	–
GSLOD1 vs. GSLOD3	S	S	S	D	–	–	–	–	...	–	–
GLOD1 vs. GLOD1	S	S	S	D	–	–	–	–	...	–	–
SLOD1 vs. SLOD1	S	S	S	S	S	S	S	S	...	S	S
GSLOD1 vs. GSLOD1	S	S	S	S	S	S	S	S	...	D	–

D/S : stands for the answer whether participant finds the test stimulus different from the reference stimulus. "D" indicates different and "S" indicates same/similar.

Table 6.3: A sample table for logging the answers of each participant

6.2.5 Analysis and evaluation

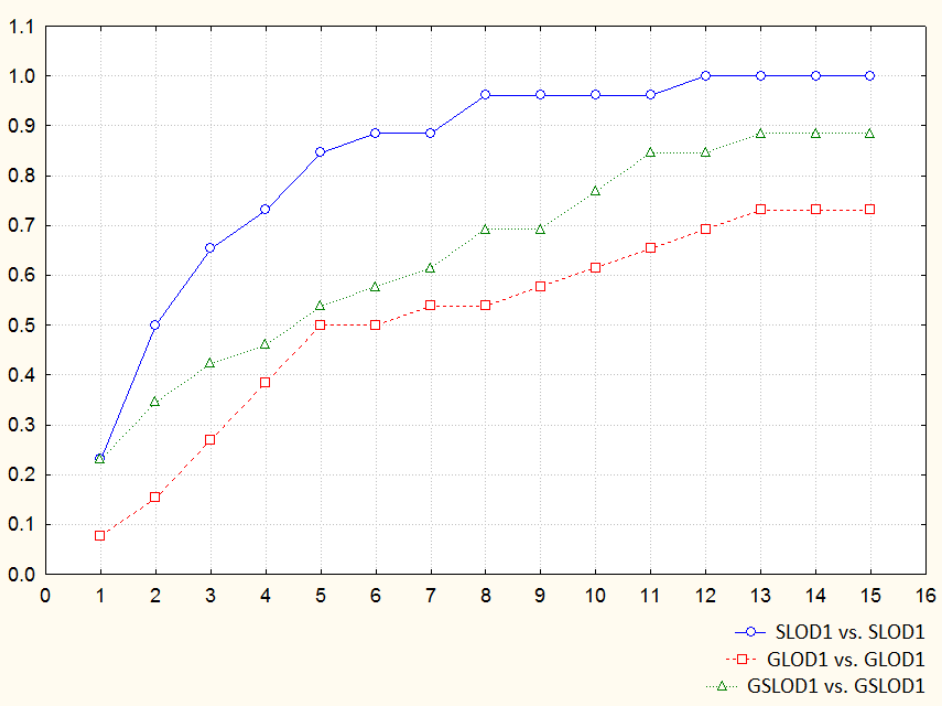
For each condition, we obtain the percentage of people who find difference between the reference stimulus and test stimulus at each stimulus intensity. By comparing the percentages between selected conditions, we will see the interaction between different LODs in auditory or/and visual modalities. And based on these percentages data, we can also find the discrimination thresholds for each condition on which we may also evaluate the capacity of crossmodal LOD in the sense of perceptual study.

In the following sections, we will infer the perception–related facts on crossmodal LOD based on data obtained from the experiment, by comparing data between test conditions.

6.2.5.1 Identical stimulus

First, we show the results of three conditions of identical stimuli in Figure 6.10. We see that, for all the conditions of identical stimulus, most participants insist on always finding difference from a certain stimulus intensity. This might explain that some kind of false memory/recognition [Brainerd and Reyna 2005] exists in our case.

Figure 6.10: Discrimination for the conditions of identical stimulus.



y-axis is the percentage of people from 0 (0%) to 1.0 (100%) who find difference. x-axis is the stimulus intensities. On the x-axis, the label 1 to 14 indicate the 14 distances (stimulus intensity) from far to near; label 15 indicates the percentage of people who find the difference at any stimulus during the whole procedure of one condition. The curve with circles (in blue) is the result of SLOD1 vs. SLOD1; the curve with squares (in red) is the result of GLOD1 vs. GLOD1; the curve with triangles (in green) is the result of GSLOD1 vs. GSLOD1.

Figure 6.10 shows that all (100%) participants found differences between identical stimuli somehow in the condition *SLOD1 vs. SLOD1*, while 73.08% and 88.46% of the participants found difference in the condition *GLOD1 vs. GLOD1* and *GSLOD1 vs. GSLOD1*, respectively. This unexpected phenomenon might be due to the difficulty of clarifying and memorising such animations (leaves movement and sound), and it seems that the difficulty for sound is more obvious than that for graphics in our case. This difficulty may be the reason that people made source monitoring error [Johnson et al. 1993], leading to the false alarms of discrimination.

The ascendant curves indicate that even in the case of identical stimulus, people attempt to convince themselves that with increasing stimuli intensity they will eventually find the difference between some reference stimulus and test stimulus. Despite of the false memory, we can see that in the condition of GSLOD1 (*GSLOD1 vs. GSLOD1*), the percentage of people at all stimulus intensities is generally smaller than that in the condition of SLOD2 and larger than that in the condition of GLOD1. It means that perceptual discrimination becomes stronger in the audio–graphic case where sound provides stronger discrimination than graphics does, compared to the sole graphics case. In other words, the crossmodal modalities neither increase nor decrease the ability of discrimination than mono-modality, and it makes the intermediate result of both mono-modality.

6.2.5.2 LOD2 conditions

Here, we show the result of three conditions of LOD2 , i.e., *SLOD1 vs. SLOD2*, *GLOD1 vs. GLOD2*, and *GSLOD1 vs. GSLOD2* in Figure 6.11, respectively.

Figure 6.11 also shows the ascendant curves, which are expected in the coarser LOD. Specifically, from the stimulus 1 to 3 on x-axis, the GSLOD2 generally has the largest percentage than both GLOD2 and SLOD2, and its curve is closer to the curve of SLOD2. On the contrary, from the stimulus 8 to 14, the curve of GSLOD2 is closer to GLOD2. In the rest, GSLOD2 is in-between GLOD2 and SLOD2. However, at the starting stimuli, the percentages are around 50%, which make sense since they indicate the location of threshold. So, it means that around the starting stimuli where the perceptual discrimination is weak, GSLOD2 makes weaker discrimination than the other two. And then when stimulus intensity increases, GSLOD2 makes intermediate result compared with the other two. In other words, the crossmodal LOD2 does increase the perceptual discrimination.

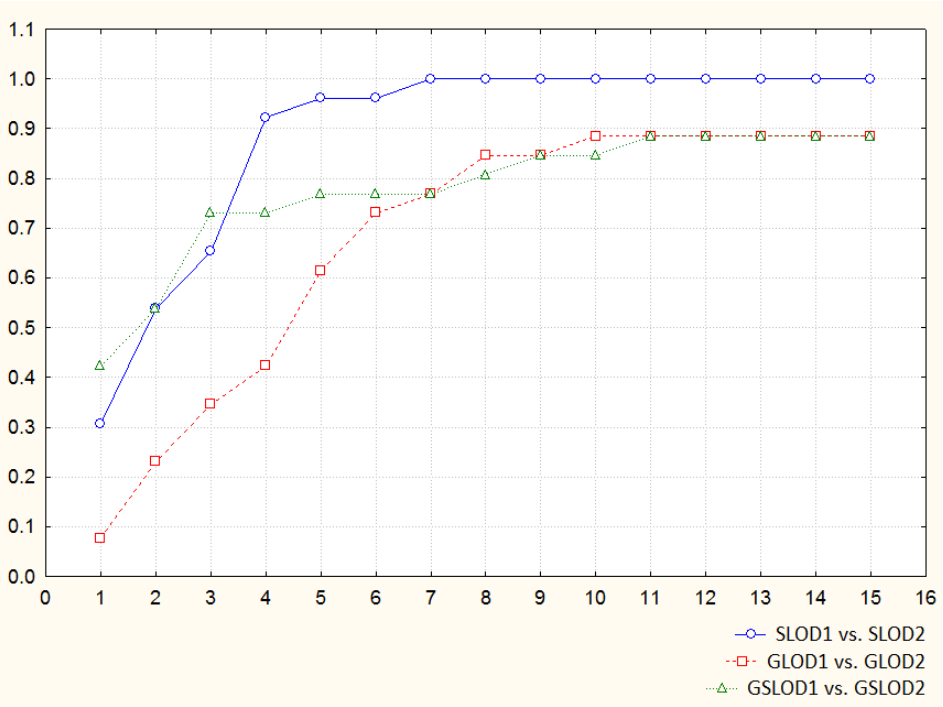


Figure 6.11: Discrimination for the conditions of LOD2.

Something interesting here is that in the condition of GLOD1/GLOD2 and also in the condition of GSLOD1/GSLOD2, there are 88.46% of participants who found difference at some stimulus intensity. This means that 11.54% of them cannot find any difference during the whole procedure.

6.2.5.3 LOD3 conditions

The discrimination result for conditions of LOD3 is given in Figure 6.12. Clearly, the curve of GSLOD3 is closer to SLOD3, and GSLOD3 also has the largest percentage than those of SLOD3 and GLOD3 at starting stimulus. So, the same conclusion as that in the LOD2 conditions can be drawn, i.e., the crossmodal LOD3 will increase the perceptual discrimination when compared with monomodal GLOD3 and monomodal SLOD3.

Note that the discrimination is interpreted as a measurement of perceived quality. One can see that the GSLOD does not provide a higher quality than monomodal LOD. This conclusion is a contrary to the result of [Bonneel et al. 2010] which are specifically concerned about perceived material quality.

We also provide the box plot for the stimulus intensity (see Figure 6.13). The

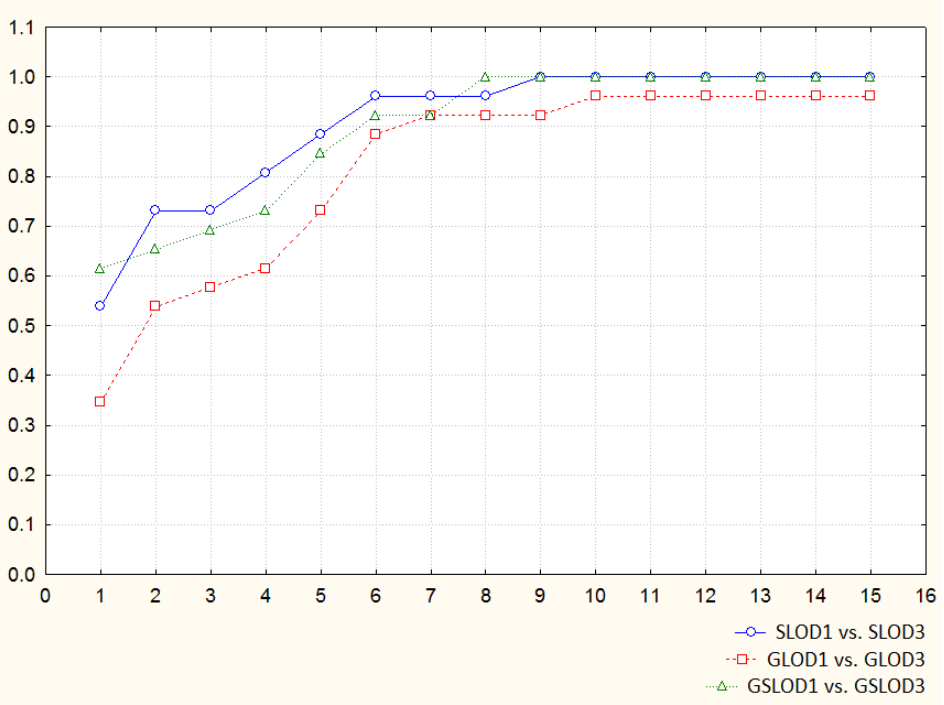


Figure 6.12: Discrimination for the conditions of LOD3.

median indicates the average stimulus intensity at which participants begin to find the difference between reference stimulus and test stimulus. One can see that the box plot also indicates the same observation as we noticed above.

6.2.5.4 Thresholds

From the precise percentage result for all stimuli in Figure 6.14 (the table), we can identify some thresholds for LODs. In Figure 6.14, the first column signifies the 14 distances and the case that one can find difference at any stimulus for a condition. C1 to C9 refer to the conditions *SLOD1 vs. SLOD1*, *GLOD1 vs. GLOD1*, *GSLOD1 vs. GSLOD1*, *SLOD1 vs. SLOD2*, *GLOD1 vs. GLOD2*, *GSLOD1 vs. SLOD2*, *SLOD1 vs. SLOD3*, *GLOD1 vs. GLOD3*, and *GSLOD1 vs. GSLOD3*, respectively.

Suppose that the practical probability for the threshold is 50%, we can see from Figure 6.14 that the threshold of GSLOD is always at lower stimulus intensity, compared to those of GLOD and SLOD.

- The threshold of GLOD2 could be between the stimulus 5 (with probability 61.54%) and the stimulus 4 (with 42.31%). The threshold of SLOD2 could

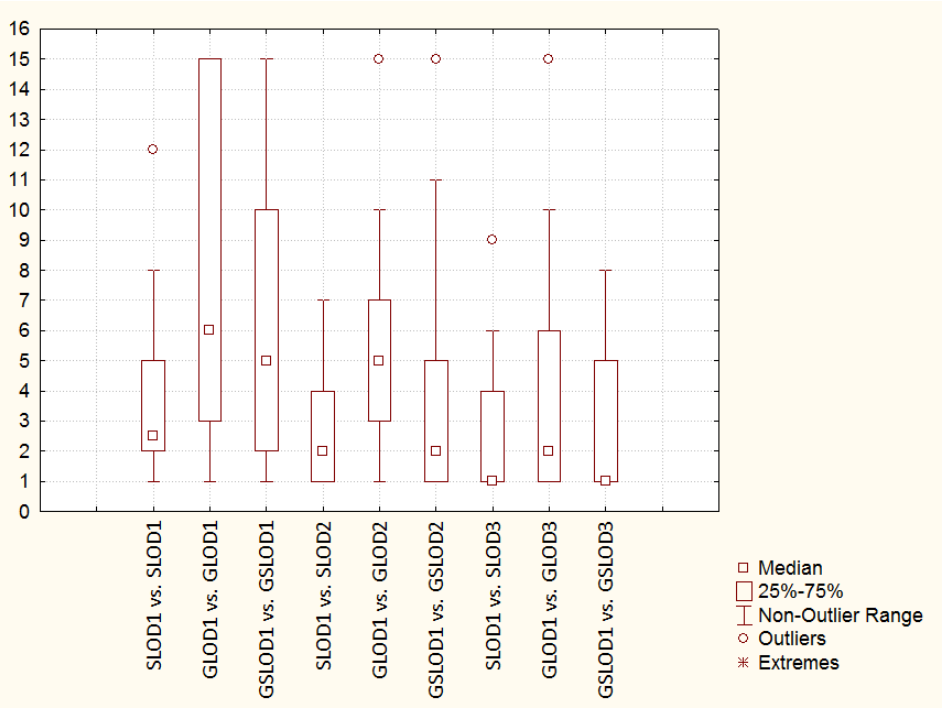


Figure 6.13: Box plot for the 9 conditions

be the stimulus 2 (with 53.85%). And the threshold of GSLOD2 could be between the stimulus 2 (with 53.85%) and the stimulus 1 (with 42.31%). It shows that GSLOD2 increases the perceptual discrimination when compared with individual GLOD2 and SLOD2.

- The threshold of GLOD3 could be the stimulus 2 (with probability 61.54%). The threshold of SLOD3 could be the stimulus 1 (with probability 53.85%). However, the threshold of GSLOD3 is not detected in the experiment. Similar to LOD2, the variation of threshold of GLOD3/SLOD3/GSLOD3 implies the same tendency that GSLOD3 increases the perceptual discrimination.

6.2.5.5 SLOD & GLOD & GSLOD conditions

Figure 6.15 shows the three discrimination curves of *SLOD1 vs. SLOD1*, *SLOD1 vs. SLOD2*, and *SLOD1 vs. SLOD3* (all the conditions of SLOD), while Figure 6.16 and Figure 6.17 show the discrimination curves of GLOD and GSLOD. Results show that LOD1 (circles), LOD2 (squares), and LOD3 (triangles) in the same modality(ies) are mostly covered one by one sequentially, which means that the curve with squares is under the one with triangles, and the curve with circles is under all the

Figure 6.14: Percentage numbers for every stimuli.

	C1	C2	C3	C4	C5	C6	C7	C8	C9
1	23,08%	7,69%	23,08%	30,77%	7,69%	42,31%	53,85%	34,62%	61,54%
2	50,00%	15,38%	34,62%	53,85%	23,08%	53,85%	73,08%	53,85%	65,38%
3	65,38%	26,92%	42,31%	65,38%	34,62%	73,08%	73,08%	57,69%	69,23%
4	73,08%	38,46%	46,15%	92,31%	42,31%	73,08%	80,77%	61,54%	73,08%
5	84,62%	50,00%	53,85%	96,15%	61,54%	76,92%	88,46%	73,08%	84,62%
6	88,46%	50,00%	57,69%	96,15%	73,08%	76,92%	96,15%	88,46%	92,31%
7	88,46%	53,85%	61,54%	100,00%	76,92%	76,92%	96,15%	92,31%	92,31%
8	96,15%	53,85%	69,23%	100,00%	84,62%	80,77%	96,15%	92,31%	100,00%
9	96,15%	57,69%	69,23%	100,00%	84,62%	84,62%	100,00%	92,31%	100,00%
10	96,15%	61,54%	76,92%	100,00%	88,46%	84,62%	100,00%	96,15%	100,00%
11	96,15%	65,38%	84,62%	100,00%	88,46%	88,46%	100,00%	96,15%	100,00%
12	100,00%	69,23%	84,62%	100,00%	88,46%	88,46%	100,00%	96,15%	100,00%
13	100,00%	73,08%	88,46%	100,00%	88,46%	88,46%	100,00%	96,15%	100,00%
14	100,00%	73,08%	88,46%	100,00%	88,46%	88,46%	100,00%	96,15%	100,00%
15	100,00%	73,08%	88,46%	100,00%	88,46%	88,46%	100,00%	96,15%	100,00%

C1 to C9 refer to the 9 conditions demonstrated in Figure 6.13, and 1 to 15 indicate the 14 distances and the case for finding difference at any stimulus. The highlighted cells indicate where the thresholds are approaching around.

others. It implies that our SLOD/GLOD/GSLOD generation is valid: the higher LOD is used, the higher quality is perceived in all the three LOD (SLOD, GLOD, GSLOD).

However, for conditions of SLOD, three curves are quite close to each other and have a few intersections, while for the other two (GLOD and GSLOD), the three curves are generally separate. It means that SLODs make less difference of discrimination among levels and are perceived with more similar qualities when compared with GLOD and GSLOD.

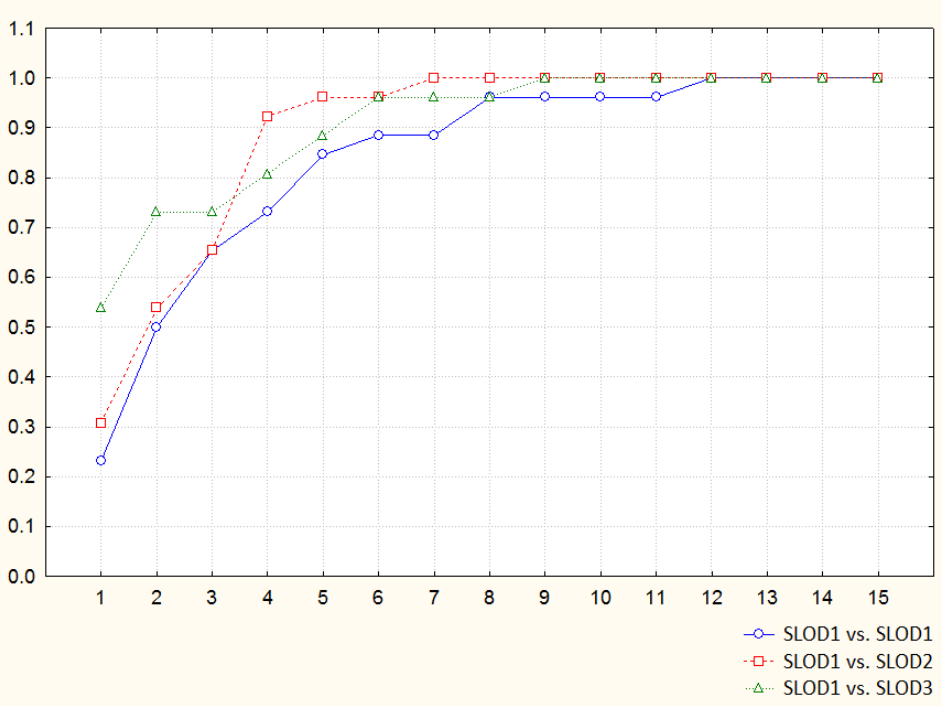


Figure 6.15: Percentages obtained in all the SLOD conditions

6.2.6 Conclusion

Different to the impostor–based method that we have employed in Chapter 4, our impostors in Approach I are not the captured images from screen of last GLOD, but the textures predefined in stock. As a result, there are no parallax errors/artifacts (see Section 1.1.1) contrary to the cases of Chapter 4. However, due to the different textures and quadrilaterals regrouping, there are differences of foliage appearance between GLODs. Such differences are visible to viewer in the transitions between two GLODs, so it is called transitional errors (see Section 1.1.1). In other words, in

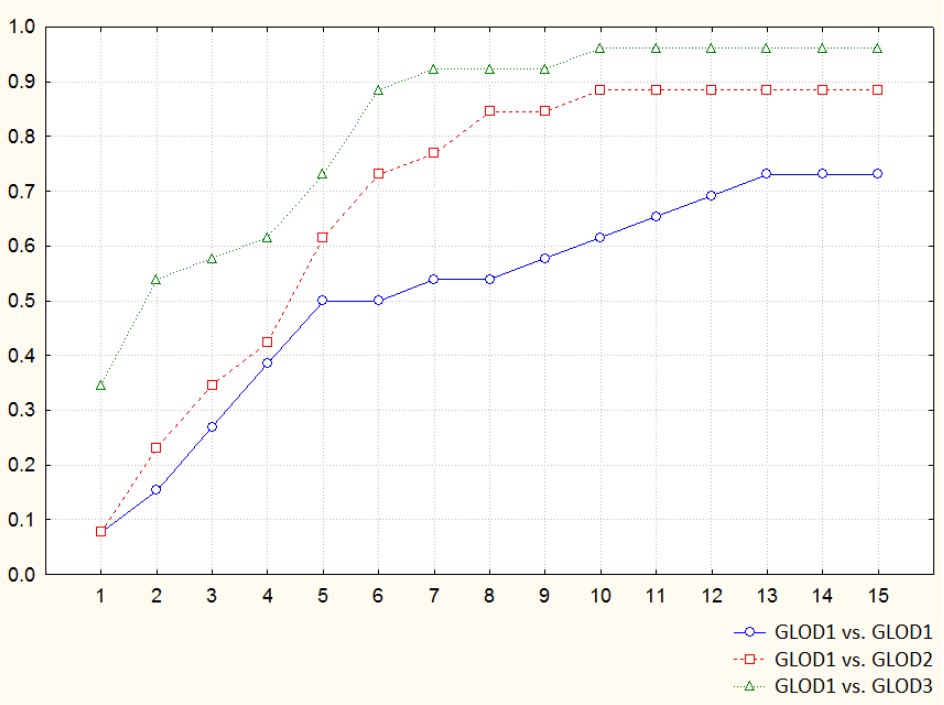


Figure 6.16: Percentages obtained in all the GLOD conditions

our experiment participants did not perceive any visual artifacts but are only asked to judge the difference between a reference and an LOD in mono- or bi-modality. To conclude, our results show that, contrary to existing works, crossmodal GSLOD does not improve perceived quality when compared with GLOD or SLOD. That means crossmodal GSLOD needs to be upgraded in detail to keep the same perceived quality than GLOD or SLOD alone. This could be a guideline for audio–graphics applications when developers use a crossmodal LOD for optimization purposes. Also, we have introduced a guideline and the methodology for generating a crossmodal discrete GSLOD system. In our system, SLOD and GLOD have a different result on perceptual discrimination. In the future, we will improve the sound and graphics to have them at a similar level of discrimination. We will also improve the GSLOD generation and selection by using a perceptually driven approach.

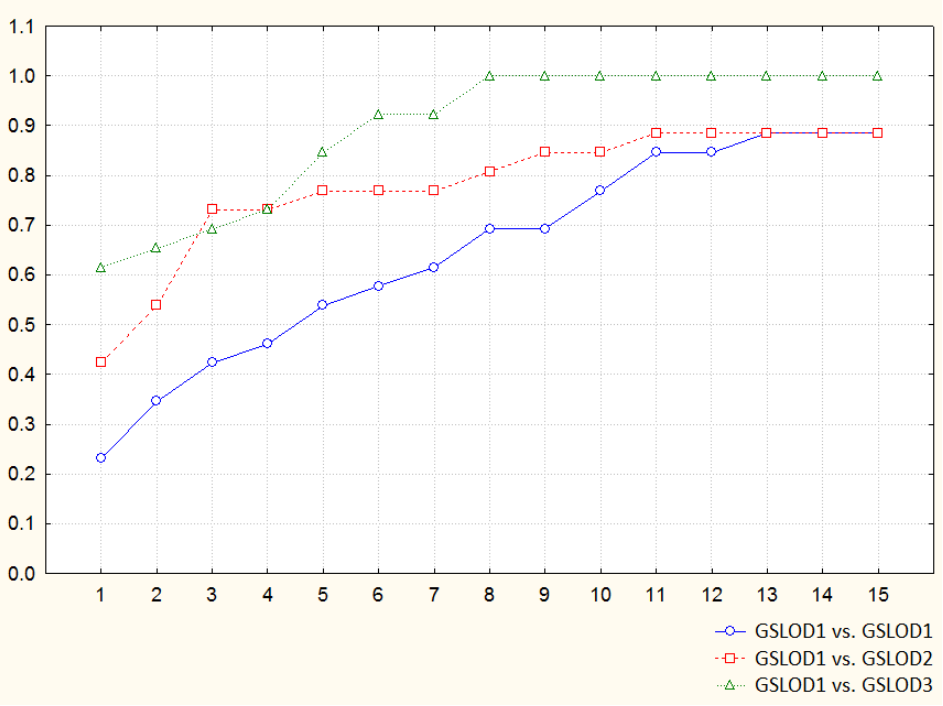


Figure 6.17: Percentages obtained in all the GSLOD conditions

Thesis conclusion

Conclusion in general

This thesis is a scientific doctoral report, not only presenting our contributions in the field, but also discussing our thinking and methods in the problem solving. We focused on the questions of audio–graphic LOD evaluation, and presented our experimental investigation. The results provide a guideline for practitioners who wish to design and render an audio–visual scene with LOD. We also provided an innovative method for audio–graphic modeling in a complex context. Such a method is represented in a standard XML–based format, X3D, for a more general usage of sharing and exchanging 3D audio–graphic data.

We would like to list the main research questions addressed in our research, which have been raised in Section 1.2.5, in order to conclude our work and summarize our contributions with respect to these issues:

Can user perception of artifacts due to impostor–based LOD be affected by a simulated, informative sound? How?

We attempted to answer the question by assessing the visual ability of artifact detection based on an impostor–based LOD rendering, with and without simulated sound. Based on our development of an impostor–based LOD scene which brings discontinuity *artifacts*, we designed a perceptual experiment of a detection task to investigate the impact of audio modality on visual perception/detection of artifacts due to impostor–based LOD rendering.

The results show that the simple stereo sound simulation can only give a more realistic feeling for visual perception when no visual artifacts is perceived. But the sound slightly aggravates the perception of discontinuity artifacts. The former finding implies that during impostor–based LOD selection, the graphical rendering of objects should be less degraded when a simulated sound of scene is added, when compared to the case when there is no sound.

Can user perception of artifacts due to impostor–based LOD be affected by a simulated, informative sound at a different level of detail? How?

We developed a realistic audio–graphic tree scene with wind-based motion and sound generation. Two separate but dependent engines were designed for rendering graphics and sound, respectively. One is *Graphics Engine* for graphics rendering to support impostor–based GLOD selection; the other is *Audio Engine* for sound rendering and impostor–based SLOD selection. Based on these systems, we used psychophysical methods for designing perceptual experiments to evaluate the impact of GSLOD on interactive rendering.

Our results shows that, crossmodal GSLOD does not improve the perceived quality when compared with GLOD or SLOD, which means that crossmodal GSLOD needs to be upgraded in detail to keep the same perceived quality than using GLOD or SLOD alone. We suggest to consider it as a guideline for audio–graphics applications when developers apply a crossmodal LOD to optimize systems.

If there is a relationship between visual LOD and auditory LOD, can it be applied to control rendering parameters for LOD selection?

According to the results we obtained from the perceptual evaluations of GSLOD, we strongly believe that effective empirical evidences can provide a method to generate a predictor function, so that an automatic measure based on crossmodal perception can be applied in run time audio–graphic LOD rendering.

In addition to answering these research questions, we have also explored the audio–graphic modeling and representation method. We have produced new knowledge and further understanding of audio–graphic modeling framework through an XML based formalism. We presented not only the principles of our modeling and representation methods, but also provided the corresponding XML Schema design.

Perspectives

First of all, we have not been able to study the perception of all kinds of artifacts based on image–based LOD rendering through multi–modality. The future work

can be the evaluation of ability of perception of different kind of artifacts, such as popping effect and blurring. Also there are other simplification forms for GLOD which we can continue to integrate with SLOD. The conventional psychophysical methods can be always improved according to specific requirement, and one can work on the improvement based on the user experience.

On the other hand, we have implemented and examined one specific scenario, the tree. There are also other scenes that are interesting to be studied, such as raining, traffic, etc.

Finally, due to the complexity of auditory events under different conditions, we have not realized all situations or scenarios for completing our modeling method. Nevertheless, thanks to the extensionality of X3D, our scheme can be always completed and enriched as long as our modeling method is needed to be improved.

Bibliography

Woodrow Barfield, Claudia M. Hendrix, Ove Bjorneseth, Kurt A. Kaczmarek, and Wouter Lotens. Comparison of human sensory capabilities with technical specifications of virtual environment equipment. *Presence*, pages 329–356, 1995. 36

N. Bonneel, G. Drettakis, N. Tsingos, I. Viaud-Delmon, and D. James. Fast modal sounds with scalable frequency-domain synthesis. *ACM Transactions on Graphics (TOG)*, 27(3):24, 2008. 82, 94

Nicolas Bonneel, Clara Suied, Isabelle Viaud-Delmon, and George Drettakis. Bimodal perception of audio-visual material properties for virtual environments. *ACM Trans. Appl. Percept.*, 7:1:1–1:16, January 2010. ISSN 1544-3558. doi: http://doi.acm.org/10.1145/1658349.1658350. URL `http://doi.acm.org/10.1145/1658349.1658350`. 16, 29, 37, 44, 55, 90, 131

Tifanie Bouchara, Christian Jacquemin, and Katz Brian F.G. Cueing multimedia search with audio-visual blur. *ACM Transactions on Applied Perception*, 2013. 36

Charles J. Brainerd and Valerie F. Reyna. *The Science Of False Memory*. Oxford University Press, USA, 1st edition, 2005. 129

James H. Clark. Hierarchical geometric models for visible surface algorithms. *Communications of the ACM*, 19(10):547–554, October 1976. ISSN 0001-0782. 23

Jonathan Cohen, Amitabh Varshney, Dinesh Manocha, Greg Turk, Hans Weber, Pankaj Agarwal, Frederick Brooks, and William Wright. Simplification envelopes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, pages 119–128, New York, NY, USA, 1996. ACM. ISBN 0-89791-746-4. doi: 10.1145/237170.237220. URL `http://doi.acm.org/10.1145/237170.237220`. 31

FrancisB. Colavita. Human sensory dominance. *Perception & Psychophysics*, 16(2): 409–412, 1974. ISSN 0031-5117. 35

Douglas Cunningham and Christian Wallraven. *Experimental Design: From User Studies to Psychophysics*. Taylor & Francis, November 2011a. ISBN 9781568814681. 17

D.W. Cunningham and C. Wallraven. *Experimental Design: From User Studies to Psychophysics*. An A K Peters book. CRC Press, 2011b. ISBN 9781568814681. URL `http://books.google.fr/books?id=rEIpkgAACAAJ`. 59, 61

Xavier Décoret, Gernot Schaufler, François X. Sillion, and Julie Dorsey. Multi-layered impostors for accelerated rendering. In *Computer Graphics Forum (Proc. of Eurographics '99)*, Grenade, Espagne, 1999. 28

Xavier Décoret, Frédo Durand, François X. Sillion, and Julie Dorsey. Billboard clouds for extreme model simplification. *ACM Trans. Graph.*, 22(3):689–696, July 2003. ISSN 0730-0301. 28, 30

Michael Deering. Geometry compression. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '95, pages 13–20, New York, NY, USA, 1995. ACM. ISBN 0-89791-701-4. doi: 10.1145/218380.218391. URL `http://doi.acm.org/10.1145/218380.218391`. 29

Hui Ding, Diemo Schwarz, Christian Jacquemin, and Roland Cahen. Spatial audio–graphic modeling for x3d. In *Proceedings of the Sixteenth Annual International Conference on 3D Web Technology*, 2011. 9, 93, 94, 95, 99, 109

Jon Driver and Charles Spence. Crossmodal attention. *Current Opinion in Neurobiology*, 8(2):245 – 253, 1998. ISSN 0959-4388. doi: 10.1016/S0959-4388(98)80147-5. URL `http://www.sciencedirect.com/science/article/pii/S0959438898801475`. 36

Reynald Dumont, Fabio Pellacini, and James A. Ferwerda. A perceptually-based texture caching algorithm for hardware-based rendering. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, pages 249–256, London, UK, UK, 2001. Springer-Verlag. ISBN 3-211-83709-4. 29

D.G. Elmes, B.H. Kantowitz, and III Henry L. Roediger. *Research Methods in Psychology*. Wadsworth Cengage Learning, 2011. ISBN 9781111350741. URL http://books.google.fr/books?id=Mz1rK_vqzt4C. 56

G.T. Fechner. *Elemente der Psychophysik*. Number v. 2 in Elemente der Psychophysik. Breitkopf und Härtel, 1860. URL http://books.google.fr/books?id=oX4NAAAAYAAJ. 59

James A. Ferwerda. Psychophysics 101: how to run perception experiments in computer graphics. In *ACM SIGGRAPH 2008 classes*, SIGGRAPH '08, pages 87:1–87:60, New York, NY, USA, 2008. ACM. doi: 10.1145/1401132.1401243. URL http://doi.acm.org/10.1145/1401132.1401243. 64

Adrian Freed, John MacCallum, Andy Schmeder, and David Wessel. Visualizations and interaction strategies for hybridization interfaces. In *Proceedings of the International Conference for New Instruments for Musical Expression NIME*, pages 343–347, 2010. 99

Thomas A. Funkhouser and Carlo H. Séquin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '93, pages 247–254, New York, NY, USA, 1993. ACM. ISBN 0-89791-601-8. 32

Thomas A. Funkhouser, Patrick Min, and Ingrid Carlbom. Real-time acoustic modeling for distributed virtual environments. In *Proceedings of SIGGRAPH 99*, pages 365–374, August 1999. 82, 94

Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '97, pages 209–216, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co. ISBN 0-89791-896-7. doi: 10.1145/258734.258849. URL http://dx.doi.org/10.1145/258734.258849. 27, 31

George A. Gescheider. *Psychophysics: The Fundamentals*. 3rd revised edition edition, 1997. ISBN 080582281X. 124, 127

T.S. Gieng, B. Hamann, K.I. Joy, G.L. Schussman, and I.J. Trotts. Constructing hierarchies for triangle meshes. *Visualization and Computer Graphics, IEEE*

Transactions on, 4(2):145–161, 1998. ISSN 1077-2626. doi: 10.1109/2945.694956. 25

David Grelaud, Nicolas Bonneel, Michael Wimmer, Manuel Asselot, and George Drettakis. Efficient and practical audio-visual rendering for games using cross-modal perception. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, 2009. URL `http://www-sop.inria.fr/reves/Basilic/2009/GBWAD09`. 82

Bernd Hamann. A data reduction scheme for triangulated surfaces. *Computer Aided Geometric Design*, 11(2):197 – 214, 1994. ISSN 0167-8396. 25

Taosong He, Lichan Hong, Amitabh Varshney, and Sidney Wang. Controlled topology simplification. *IEEE Transactions on Visualization and Computer Graphics*, 2:171–184, 1996. 27

S. Hidaka, Y. Manaka, W. Teramoto, Y. Sugita, R. Miyauchi, J. Gyoba, Y. Suzuki, and Y. Iwaya. Alternation of sound location induces visual motion perception of a static object. *PLoS ONE*, 4:8188, dec 2009. 36

Lewis E. Hitchner and Michael W. Mcgreevy. Methods for user-based reduction of model complexity for virtual planetary exploration. *Proceedings of the SPIE, The International Society for Optical Engineering*, 1913, 1993. 32

H. Hoppe. Progressive meshes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 99–108. ACM, 1996. ISBN 0897917464. 13, 25, 31

H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 19–26. ACM, 1993. ISBN 0897916018. 25

ITU. Methodology for the subjective assessment of quality of television pictures. In *Rec. BT. 500*, 2009. 126

M K Johnson, S Hashtroudi, and D S Lindsay. Source monitoring. *Psychological Bulletin*, 114:3–28, July 1993. 130

B.H. Kantowitz, III Henry L. Roediger, and D.G. Elmes. *Experimental psychology*. International student edition. Wadsworth Cengage Learning, 2009. ISBN 9780495595335. URL `http://books.google.fr/books?id=2-5VL8PHLsIC`. 56

Ladislav Kavan, Simon Dobbyn, Steven Collins, Jiří Žára, and Carol O'Sullivan. Polypostors: 2d polygonal impostors for 3d crowds. In *Proceedings of the 2008 symposium on Interactive 3D graphics and games*, I3D '08, pages 149–155, New York, NY, USA, 2008. ACM. ISBN 978-1-59593-983-8. doi: 10.1145/1342250.1342273. URL http://doi.acm.org/10.1145/1342250.1342273. 27

Pontus; Vastfjall Daniel; Kleiner Mendel Larsson. Better presence and performance in virtual environments by improved binaural sound rendering. In *Audio Engineering Society Conference: 22nd International Conference: Virtual, Synthetic, and Entertainment Audio*, 6 2002. URL http://www.aes.org/e-lib/browse.cfm?elib=11148. 82

Aristid Lindenmayer. Developmental algorithms for multicellular organisms: A survey of l-systems. *Journal of Theoretical Biology*, 54(1):3 – 22, 1975. ISSN 0022-5193. doi: http://dx.doi.org/10.1016/S0022-5193(75)80051-8. URL http://www.sciencedirect.com/science/article/pii/S0022519375800518. 74

Michael Lippert, Nikos K. Logothetis, and Christoph Kayser. Improvement of visual contrast detection by a simultaneous sound. *Brain Research*, 1173:102–109, 2007. 36

David Luebke and Carl Erikson. View-dependent simplification of arbitrary polygonal environments. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '97, pages 199–208, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co. ISBN 0-89791-896-7. doi: 10.1145/258734.258847. URL http://dx.doi.org/10.1145/258734.258847. 26, 31

David Luebke, Benjamin Watson, Jonathan D. Cohen, Martin Reddy, and Amitabh Varshney. *Level of Detail for 3D Graphics*. Elsevier Science Inc., New York, NY, USA, 2002. ISBN 1558608389. 13, 17, 25, 26, 30, 32, 33, 56, 59, 113

Paulo W. C. Maciel and Peter Shirley. Visual navigation of large environments using textured clusters. In *Proceedings of the 1995 symposium on Interactive 3D graphics*, I3D '95, pages 95–ff., New York, NY, USA, 1995. ACM. ISBN 0-89791-736-7. doi: 10.1145/199404.199420. 28

Nelson Max. Hierarchical rendering of trees from precomputed multi-layer z-buffers.

In *Proceedings of the eurographics workshop on Rendering techniques '96*, pages 165–174, London, UK, UK, 1996. Springer-Verlag. ISBN 3-211-82883-4. 28

Alex Mohr and Michael Gleicher. Deformation sensitive decimation. Technical report, University of Wisconsin Graphics Group, 2003. 27

Tomohiko Mukai and Shigeru Kuriyama. Multilinear motion synthesis with level-of-detail controls. In *Proceedings of the 15th Pacific Conference on Computer Graphics and Applications*, PG '07, pages 9–17, Washington, DC, USA, 2007. IEEE Computer Society. ISBN 0-7695-3009-5. doi: 10.1109/PG.2007.44. URL `http://dx.doi.org/10.1109/PG.2007.44`. 27

Fakir S. Nooruddin and Greg Turk. Simplification and repair of polygonal models using volumetric techniques. *IEEE Transactions on Visualization and Computer Graphics*, 9:191–205, 2003. 27

James F. O'Brien, Chen Shen, and Christine M. Gatchalian. Synthesizing sounds from rigid-body simulations. In *ACM SIGGRAPH/Eurographics Symp. on Computer animation*, pages 175–181, 2002. ISBN 1-58113-573-4. doi: http://doi.acm.org/10.1145/545261.545290. URL `http://doi.acm.org/10.1145/545261.545290`. 37, 82, 94, 112

Eric C. Odgaard, Yoav Arieh, and Lawrence E. Marks. Cross-modal enhancement of perceived brightness: Sensory interaction versus response bias. *Perception & Psychophysics*, 65:123–132, 2003. 36

Toshikazu Ohshima, Hiroyuki Yamamoto, and Hideyuki Tamura. Gaze-directed adaptive rendering for interacting with virtual space. In *Proceedings of the 1996 Virtual Reality Annual International Symposium (VRAIS 96)*, VRAIS '96, pages 103–, Washington, DC, USA, 1996. IEEE Computer Society. ISBN 0-8186-7295-1. 32

Marc Olano, Bob Kuehne, and Maryann Simmons. Automatic shader level of detail. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, HWWS '03, pages 7–14, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association. ISBN 1-58113-739-7. URL `http://dl.acm.org/citation.cfm?id=844174.844176`. 13, 29

Jovan Popović and Hugues Hoppe. Progressive simplicial complexes. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '97, pages 217–224, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co. ISBN 0-89791-896-7. doi: 10.1145/258734.258852. URL http://dx.doi.org/10.1145/258734.258852. 25

Monique Radeau and Paul Bertelson. The effect of a textured visual field on modality dominance in a ventriloquism situation. *Perception & Psychophysics*, 20(4):227–235, 1976. ISSN 0031-5117. doi: 10.3758/BF03199448. URL http://dx.doi.org/10.3758/BF03199448. 35

Martin Reddy. *Perceptually Modulated Level of Detail for Virtual Environments*. PhD thesis, University of Edinburgh, 1997. 13, 15, 24, 33, 34

E. Reinhard. *Color Imaging: Fundamentals and Applications*. Ak Peters Series. A.K. Peters, 2008. ISBN 9781568813448. URL http://books.google.fr/books?id=79-V0_TElg4C. 31

Remi Ronfard and J. Rossignac. Full-range approximation of triangulated polyhedra. *Computer Graphics Forum*, 15(3):67–76, 1996. doi: 10.1111/1467-8659.1530067. 31

Jarek Rossignac and Paul Borrel. Multi-resolution 3d approximations for rendering complex scenes. In Bianca Falcidieno and TosiyasuL. Kunii, editors, *Modeling in Computer Graphics*, IFIP Series on Computer Graphics, pages 455–465. Springer Berlin Heidelberg, 1993. ISBN 978-3-642-78116-2. doi: 10.1007/978-3-642-78114-8_29. URL http://dx.doi.org/10.1007/978-3-642-78114-8_29. 26

Gernot Schaufler, Wolfgang StÃ¼rzlinger, Johannes Kepler, Universitt Linz, and A-Linz. A three dimensional image cache for virtual reality. In *Computer Graphics Forum*, pages 227–236, 1996. 28

William J. Schroeder. A topology modifying progressive decimation algorithm. In *Proceedings of the 8th conference on Visualization '97*, VIS '97, pages 205–ff., Los Alamitos, CA, USA, 1997. IEEE Computer Society Press. ISBN 1-58113-011-2. 25

William J. Schroeder, Jonathan A. Zarge, and William E. Lorensen. Decimation of triangle meshes. *SIGGRAPH Comput. Graph.*, 26(2):65–70, jul 1992. ISSN 0097-8930. doi: 10.1145/142920.134010. URL `http://doi.acm.org/10.1145/142920.134010`. 13, 26, 27

Diemo Schwarz. Corpus-based concatenative synthesis : Assembling sounds by content-based selection of units from large sound databases. *IEEE Signal Processing*, 24-2:92–104, 2007. 39, 110

Diemo Schwarz and Norbert Schnell. Descriptor-based sound texture sampling. In *Sound and Music Computing (SMC)*, pages 510–515, Barcelona, Spain, Juillet 2010. URL `http://articles.ircam.fr/textes/Schwarz10a/`. 110

Diemo Schwarz, GrÃ©gory Beller, Bruno Verbrugghe, Sam Britton, and Ircam Centre Pompidou. Real-time corpus-based concatenative synthesis with catart. In *in Proc. of the Int. Conf. on Digital Audio Effects (DAFx-06*, pages 279–282, 2006. 39

Diemo Schwarz, Roland Cahen, Hui Ding, and Christian Jacquemin. Sound level of detail in interactive audiographic 3d scenes. In *P-ICMC*, Huddersfield, UK, 2011. 40, 49, 98, 112, 116, 118

Jonathan Shade, Dani Lischinski, David H. Salesin, Tony DeRose, and John Snyder. Hierarchical image caching for accelerated walkthroughs of complex environments. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, pages 75–82, New York, NY, USA, 1996. ACM. ISBN 0-89791-746-4. 28

Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '98, pages 231–242, New York, NY, USA, 1998. ACM. ISBN 0-89791-999-8. 28

Ladan Shams and Robyn Kim. Crossmodal influences on visual perception. *Physics of Life Reviews*, 7(3):269–284, Sep. 2010. ISSN 1571-0645. 36

C Spence and J Driver. Audiovisual links in endogenous covert spatial attention. *Journal of Experimental Psychology: Human Perception and Performance*, 22(4): 1005–1030, 1996. 36

Barry E. Stein, Nancy London, Lee K. Wilkinson, and Donald D. Price. Enhancement of perceived visual intensity by auditory stimuli: A psychophysical analysis. *J. Cognitive Neuroscience*, 8:497–506, November 1996. ISSN 0898-929X. doi: 10.1162/jocn.1996.8.6.497. URL `http://portal.acm.org/citation.cfm?id=1596855.1596858`. 36

Russell L. Storms and Maj Usa. Auditory-visual cross-modal perception. In *In ICAD*, 2000. 17, 35

Russell L. Storms and Michael J. Zyda. Interactions in perceived quality of auditory-visual displays. *Presence: Teleoper. Virtual Environ.*, 9:557–580, December 2000. ISSN 1054-7460. doi: 10.1162/105474600300040385. URL `http://dl.acm.org/citation.cfm?id=1246908.1246911`. 36

Topophonie. Project scrapbook: Topophonie research project – audiographic cluster navigation (2009–2012). Technical report, 2013. 13, 43, 46, 48, 50, 53

Nicolas Tsingos, Thomas Funkhouser, Addy Ngan, and Ingrid Carlbom. Modeling acoustics in virtual environments using the uniform theory of diffraction. In *Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH, pages 545–552, 2001. ISBN 1-58113-374-X. doi: http://doi.acm.org/10.1145/383259.383323. URL `http://doi.acm.org/10.1145/383259.383323`. 82, 94, 112

Nicolas Tsingos, Emmanuel Gallo, and George Drettakis. Perceptual audio rendering of complex virtual environments. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, pages 249–258, New York, NY, USA, 2004. ACM. doi: 10.1145/1186562.1015710. URL `http://doi.acm.org/10.1145/1186562.1015710`. 15, 16

Aleksander Valjamae and Salvador Soto-Faraco. Filling-in visual motion with sounds. *Acta Psychologica*, 129(2):249–254, 2008. 36

Kees van den Doel and Dinesh K. Pai. *Modal synthesis for vibrating objects.* Ed. Ken Greenebaum, A. K. Peter, Natick, MA, 2003. 82

Jason Williams and Jarek Rossignac. Mason: morphological simplification. *Graph. Models*, 67(4):285–303, July 2005. ISSN 1524-0703. 27

Lance Williams. Pyramidal parametrics. *SIGGRAPH Comput. Graph.*, 17(3):1–11, July 1983. ISSN 0097-8930. doi: 10.1145/964967.801126. URL `http://doi.acm.org/10.1145/964967.801126`. 29

Wanmin Wu, Ahsan Arefin, Gregorij Kurillo, Pooja Agarwal, Klara Nahrstedt, and Ruzena Bajcsy. Color-plus-depth level-of-detail in 3d tele-immersive video: a psychophysical approach. In *Proceedings of the 19th ACM international conference on Multimedia*, MM '11, pages 13–22, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0616-4. doi: http://doi.acm.org/10.1145/2072298.2072302. URL `http://doi.acm.org/10.1145/2072298.2072302`. 124, 126

Julie C. Xia, Jihad El-Sana, and Amitabh Varshney. Adaptive real-time level-of-detail-based rendering for polygonal models. *IEEE Transactions on Visualization and Computer Graphics*, 3(2):171–183, April 1997. ISSN 1077-2626. 31

Appendix A

Spatial Audio–Graphic Modeling for X3D

Hui Ding* LIMSI–CNRS–UPSUD
Diemo Schwarz† IRCAM–CNRS–UPMC
Christian Jacquemin‡ LIMSI–CNRS–UPSUD
Roland Cahen§ ENSCI–Les Ateliers

Abstract

In audio–graphic scenes, visual and audio modalities are synchronized in time and space, and their behaviour is determined by a common process. We present here a novel way of modeling audio–graphic content for interactive 3D scenes with the concept of sound processes and their activation through 2D or 3D profiles. Many 3D applications today support both graphical and audio effects to provide a more realistic user experience; however a common model and interchange format for interactive audio–graphic scenes is still lacking. X3D is one of the most promising formats for 3D scene representation. It is extensible and supports simple spatial audio representation and almost all basic and advanced 3D computer graphics techniques. We therefore propose an extension of the X3D standard to represent the sound process and activation profile model for providing a rich audio–graphic description in X3D.

Keywords: X3D, audio–graphic modeling, sound process

1 Introduction

As a very promising format for representing 3D computer graphics, X3D supports almost all the common 2D/3D graphic techniques. Besides, X3D is a functionally comprehensive format and also supports spatialized audio and video, specifically mapping the audio-visual sources onto geometry in the scene. However, the spatialized sound in X3D is in general not refined and exhausted enough to represent an interactive audio-graphic scene. For example, it is not capable of describing the variable sound of leaves through wind due to sophisticated sound activation. Therefore, we extend X3D by introducing a novel method of audio-graphic modeling.

Most prior 3D scene description languages have mainly focused on visualization and auralization separately. [Funkhouser et al. 1999] and [Tsingos et al. 2001] have computed the sound propagation paths by simulating them as wave phenomena. [O'Brien et al. 2002] and [Bonneel et al. 2008] have focused on sound synthesis from different physical motions. Meanwhile, [Tsingos et al. 2004] and [Moeck et al. 2007] have investigated the computational limit problem by using auditory culling and clustering technologies to reduce the complexity of audio rendering. However, these works did not address the issue of the description and activation of sound process in 3D graphical environments. By considering together audio and graphic cues to 3D virtual environments, we present new concepts for characterizing the principle of representing such a sound process.

To sum up, we incorporate the concepts of sound processes and their activation for modeling audio–graphic scenes into X3D by extending the conventional schema in the private-extension schema. We think that our audio-graphic modeling method is suitable to be expressed in X3D, and with X3D support, our method could become more standardized.

The rest of the paper is organized as follows. Section 2 presents the principle of 3D audio-graphic scene description, the definition of the sound process and its specification in X3D. Section 3 describes how to place the sound source in the 3D scene with respect to X3D. Section 4 presents the activation profile and its specification in X3D. Finally, Section 5 concludes the paper.

2 Audio–Graphic Modeling

Graphic and sounding objects are audio–graphic when visual and audio modalities are synchronized in time and space and when they share a common process. In an audio–graphic scene, the graphics information is often visualized to viewer, while the audio information is relatively invisible and perceived only when activated. Different from work on acoustics [Tsingos et al. 2001] or sound synthesis [O'Brien et al. 2002], our work is focused on how to place the sound sources in a 3D scene and their mapping to a sound process which can be activated by corresponding sound activation profiles according to different audio–graphic content.

Our principle of sound representation is that each sound *effector* is situated at a single point of a geometry object in the 3D scene (see section 3.1). The produced sound might occupy a larger space, but this *hotpoint* is where the sound processes activation level is measured. Each sound point is linked to a *sound process* (section 2.1). The *activation* of a sound process and other parameters are determined by *profiles* or *maps* that move in the scene (section 4).

For the example of sound made by leaves in the wind, we assume that every leaf is a sound point which is linked to a sound process, and the wind can be depicted as a kind of profile. When the wind profile moves through the leaves, the sound processes will be activated. In the following sections, we specify sound process, sound mapping, activation profile and their representation in X3D. We have demonstrated our method through this audio-graphic example scene in Unity3D[1].

2.1 Sound Process Definition

The sound process, located at one or many points in a group, is defined by the following general parameters:

identifier
: A string identifying this instance of a sound process.

model
: A string refering to the class of the sound process running this instance.

activations
: A vector of numbers between 0 and 1 giving the activation levels of the process (i.e. possibly weights for several presets determining the parameters of the process).
This could be interpreted as a generic parameter vector, also.

*e-mail: hui.ding@limsi.fr
†e-mail: schwarz@ircam.fr
‡e-mail: christian.jacquemin@limsi.fr
§e-mail: cahen@ensci.fr

[1] Unity3D is an integrated development environment for creating interactive 3D content like 3D video games. http://unity3d.com/

The concrete parameters are specific to individual sound process models, e.g. *impact material* for rain sound process models, *car speed* for traffic sound process models.

Note that the volume of the sound process is tuned by its individual sound placement in section 3.1.

2.2 X3D Representation of a Sound Process

In X3D, the AudioClip and the MovieTexture are so far the only possible sound sources (the former playing samples, MP3, or MIDI files).

We propose to extend the formalism by a new subclass of X3DSoundSourceNode that we call TPSoundProcess, that can have the attributes corresponding to the sound process parameters listed above.

The new object hierarchy is depicted in figure 1.

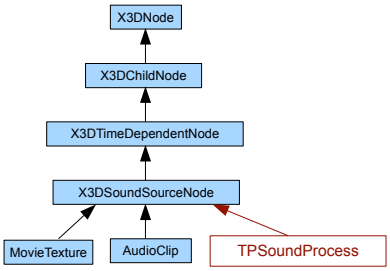


Figure 1: *The proposed element TPSoundProcess extending X3DSoundSourceNode.*

Its specification of the XML encoding is:

```
<TPSoundProcess
 DEF=""                      ID
 USE=""                      IDREF
 description=""              SFString   [inputOutput]
 loop="false"                SFBool     [inputOutput]
 pauseTime="0"               SFTime     [inputOutput]
 pitch="1.0"                 SFFloat    [inputOutput]
 resumeTime="0"              SFTime     [inputOutput]
 startTime="0"               SFTime     [inputOutput]
 stopTime="0"                SFTime     [inputOutput]
 url=""                      MFString   [inputOutput]
 model=""                    SFString   [inputOutput]
 activation=""               SFString   [inputOutput]
 containerField="children"   NMTOKEN
/>
```

Compared to Sound, TPSoundProcess contains *model* and *activation* as new attributes. The attribute DEF can be used as the parameter "*identifier*".

3 Sound Source

One sound source should be located at a single point in a local coordinate system so as to usually attach itself to a 3D primitive or object. One or more sound sources should be linked to a sound process in order to emit sound when the sound process is activated by a certain profile. We present in the following subsections how to a map sound process to sources and its representation in X3D.

3.1 Sound Placement

We place sound processes (see section 2.1) at single points in the 3D scene, which correspond to the *effector*, e.g. the tree leaf that is activated, or the impact point of a raindrop.

The produced sound might occupy a larger space, as determined by the directivity information (see figure 2), but this *hotpoint* is where the sound processes activation level is measured (see section 4). Each sound point is assigned to a *sound process* instance (see section 2.1) that runs a certain *sound process* model.

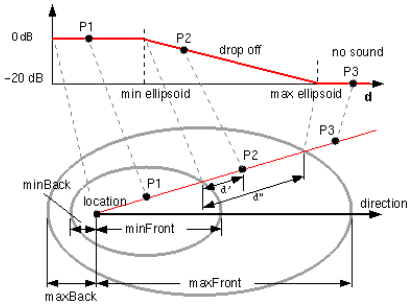


Figure 2: *X3D Sound Node Geometry (from [The Web3D Consortium 2008])*

3.2 X3D Representation of a Sound Source

In X3D, the Sound subclass of the abstract X3DSoundNode class represents the placement of a sound source at a point location and its directivity and distance-based attenuation. It was supposed to be the proper node which holds a link to our extension of X3DSoundSourceNode, which is the sound process (see section 2.1). However, since it contains only AudioClips or MovieTexture for sound playback[2], we need to create a new element named "TPSound" based on Sound (see figure 3) in order to contain our TPSoundProcess.

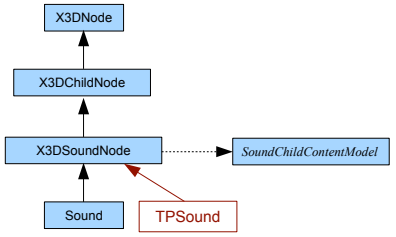


Figure 3: *The proposed element TPSound extending X3DSoundNode.*

Based on Sound, TPSound could be specified for the XML encoding as follows:

```
<TPSound
 DEF=""                      ID
 USE=""                      IDREF
 direction="0 0 1"           SFVec3f    [inputOutput]
 intensity="1"               SFFloat    [inputOutput]
 location="0 0 0"            SFVec3f    [inputOutput]
 maxBack="10"                SFFloat    [inputOutput]
 maxFront="10"               SFFloat    [inputOutput]
 minBack="1"                 SFFloat    [inputOutput]
 minFront="1"                SFFloat    [inputOutput]
 priority="0"                SFFloat    [inputOutput]
 spatialize="true"           SFBool     [initializeOnly]
 containerField="children"   NMTOKEN
 >
 <TPSoundProcess />
</TPSound>
```

Grouping of sound points to be serviced by a single sound process is achieved by assigning the same sound process instance to several sound points.

[2] See X3D encoding documentation http://www.web3d.org/x3d/specifications/ISO-IEC-19776-1.2-X3DEncodings-XML

4 Activation Profile Representation

The activation and in fact any parameter of the sound process are either determined by a global setting (a scene parameter, determined by the user or dependent on some script), or they are given by an *activation map* or *profile* [Schwarz et al. 2011]. A profile is a 2D or 3D scalar field the value of which can be looked up by position. This profile determines the activation value at each sound source point location. The profile is attached to a (possibly invisible) 2D or 3D object that can move through the scene in order to control the sound processes, e.g. wind in trees, a hand in leaves, or excitation of a crowd.

One possibility for defining the profiles detailed in the following is that of *parametric profiles* (section 4.1), where the distance to a reference point is passed through a mapping function. An alternative, not treated in this article, are *mesh-based profiles* [Freed et al. 2010], where each vertex in a point set carries an activation value, and we interpolate inside the triangular mesh.

4.1 Parametric Profiles

A parametric profile is given by a function that can be evaluated at a point in the scene. For instance, a gaussian-shaped profile has as parameters the centre, width, and curve (standard deviation, kurtosis).The function, that defines the shape of the profile, is then extended to 2D or 3D by different means detailed in section 4.1.3: revolution, inclusion in a geometric support object which defines the extent of the profile, or extrusion.

This architecture allows to compose a great variety of profile shapes from only a few primitives, and by specifying only a hand full of parameters.

4.1.1 Profile Functions

The different 1D parametric profile functions depicted in figure 4 are defined by distance to a reference point, and carry 1–4 parameters. They all have as common parameter an inversion flag. The list of functions and parameters is:

linear mindist, maxdist
: A linear segment rising from zero to one between mindist, maxdist. Beyond these, the linear function is constant at zero and one, respectively.

delta mindist, maxdist, middle, width
: trapezoidal, or cut cone shape, for fade-in/fade-out of activation

exscale mindist, maxdist, base
: Exponential curve between mindist, maxdist. Here, the *base* parameter determines the curvature of the function.

$$y = \frac{e^{\left(\log base \frac{x-mindist}{maxdist-mindist}\right)-1}}{base-1} \qquad (1)$$

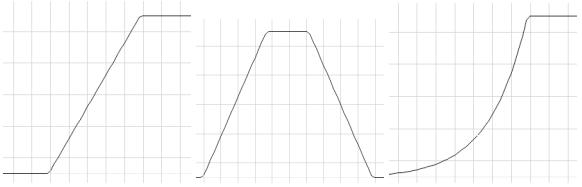


Figure 4: *The linear, delta, and exscale parametric profile function shapes.*

4.1.2 X3D representation of profile functions

We introduce a new complextype tpprofilefunction that can be inherited by x3dnode, and this tpprofilefunction will derive different concrete elements such as tpprofilefunctionlinear, tpprofilefunctiondelta, and tpprofilefunctionexscale which define the profile function (see figure 5).

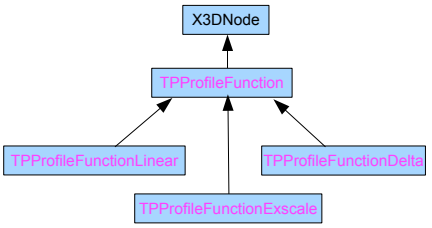


Figure 5: *Profile function class hierarchy.*

4.1.3 Profiles in 2D

Any of the 1D parametric profile functions are extended to define an activation profile in 2D (that is usually parallel to the ground) in several ways:

revolution: by revolution of the profile around the vertical y-axis at a reference origin point o, either on
: **infinite support:** purely based on the distance to o, or on
: **finite support:** i.e. the distance to o is rescaled to the distance r to the intersection point of the ray from o through the query point with the boundary of the support. then, maxdist defines the relative boundary, i.e. $scale = r/maxdist$. the finite support can be one of:
  - circular support, possibly transformed (ellipse)
  - polygonal support, either given by a list of points, or by the convex hull of a point set

extrusion: linear extension of one 1D function along a line, resulting in a rectangular or parallelogram-shaped 2D profile

interpolation: two or more 1D functions are placed in parallel in a rectangle and extruded while interpolating between neighbouring functions.

For easier authoring of 3D scenes, we can visualise the 2D profiles in 2D either by colour or transparency, or we can generate a 3D visualisation object. Here, each point of the mesh of the visualising object is projected to the ground plane and then samples the profile, affecting the profile value times a max height to the y coordinate. Figure 6 shows such a visualisation for three revolution profiles.

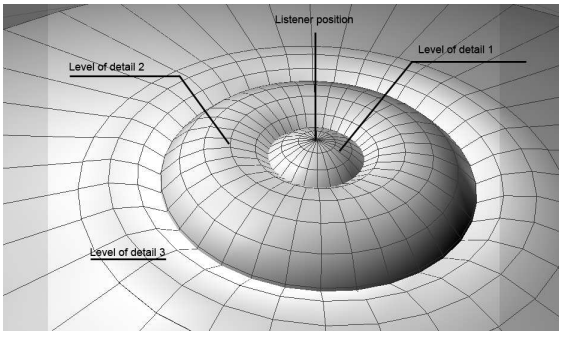


Figure 6: *3D visualisation of three revolution profiles, here placed around the listener position for control of level of detail.*

4.1.4 Profiles in 3D

The extension of the 1D parametric profile functions to 3D is similar to the 2D extension:

revolution: spherical revolution of the function around an origin o in order to create a (distance based) spherical profile, again either on
- **infinite support:** resulting in a transformed sphere (ovoid), or on
- **finite support:** deformed by a (convex) mesh, similar to the inscription into a 2D boundary in section 4.1.3.

extrusion: the 1D profile function is extruded along a plane to create a parallelepiped.

interpolation: two or more planes of 2D profiles are interpolated along a line.

4.1.5 X3D Representation of Parametric Profiles in 2D or 3D

The representation needs to link a profile node to a profile function, and a support geometric object (ellipse, polygon, cylinder, sphere, mesh), and possibly a visualisation geometric object (a mesh sampling the profile, a 2D or 3D colour texture).

A profile is attached to a geometrical object which stands for a container of the profile. This container could be 2D or 3D, and can move around in our 3D scene. In our extension X3D schema (figure 7), we add a TPGeometryProfileContainer element that will contain two child nodes corresponding respectively to X3DGeometryNode and TPProfileFunction. The group ShapeChildContentModel allows the container to be any existing 2D or 3D geometry node or to create a new geometry instance by IndexedFaceSet for example.

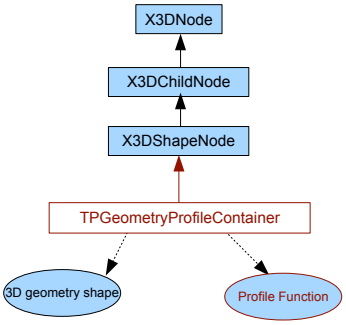


Figure 7: *TPGeometryProfileContainer class hierarchy*

The TPProfileFunctionModel in TPGeometryProfileContainer works as the SoundChildContentModel in Sound, so that TPProfileFunctionModel contains all the models of functions of profile which are classified by dimension (TPProfileFunctionModel1D, TPProfileFunctionModel2D, TPProfileFunctionModel3D). Every model contains different profile functions (for example, TPProfileFunctionModel1D contains TPProfileFunctionLinear, TPProfileFunctionDelta, TPProfileFunctionExscale).

The TPGeometryProfileContainer could be specified for the XML encoding as follows:

```
<TPGeometryProfileContainer
 DEF=""                      ID
 USE=""                      IDREF
 bboxCenter="0 0 0"          SFVec3f    [initializeOnly]
 bboxSize="-1 -1 -1"         SFVec3f    [initializeOnly]
 containerField="children"   NMTOKEN
 >
 <ShapeChildContentModel />
 <TPProfileFunctionModel />
</TPGeometryProfileContainer>
```

5 Discussion and Conclusion

Our work shows a model for audio–graphic scene description which uses the concept of sound process and its representation. To standadize our model and port it to different platforms, we would like to incorporate it into the X3D format. Compared to the traditional sound elements in X3D, our representation of sonic objects requires a much more complex information-exchange between audio and graphical processes at the rendering level. However, X3D does not directly support complex 3D computer graphics computations, thus these computations must be carried out by the scripting engine. This will be an obstacle for the X3D audio–graphic scene designer who is neither expert in computer graphics, nor in scripting languages.

In the near future, we will continue to explore further profile activation for different sound process in order to propose a complete extension schema of X3D.

Acknowledgements

This work is funded by the *Agence Nationale de la Recherche* within the project *Topophonie*[3], ANR-09-CORD-022. We thank the project partners for the common work and fruitful discussions.

References

BONNEEL, N., DRETTAKIS, G., TSINGOS, N., VIAUD-DELMON, I., AND JAMES, D. 2008. Fast modal sounds with scalable frequency-domain synthesis. *ACM Transactions on Graphics (TOG) 27*, 3, 24.

FREED, A., MACCALLUM, J., SCHMEDER, A., AND WESSEL, D. 2010. Visualizations and Interaction Strategies for Hybridization Interfaces. In *Proceedings of the International Conference for New Instruments for Musical Expression NIME*, 343–347.

FUNKHOUSER, T. A., MIN, P., AND CARLBOM, I. 1999. Real-time acoustic modeling for distributed virtual environments. In *Proceedings of SIGGRAPH 99*, 365–374.

MOECK, T., BONNEEL, N., TSINGOS, N., DRETTAKIS, G., VIAUD-DELMON, I., AND ALOZA, D. 2007. Progressive perceptual audio rendering of complex scenes. In *ACM SIGGRAPH Symp. on Interactive 3D Graphics and Games*.

O'BRIEN, J. F., SHEN, C., AND GATCHALIAN, C. M. 2002. Synthesizing sounds from rigid-body simulations. In *ACM SIGGRAPH/Eurographics Symp. on Computer animation*, 175–181.

SCHWARZ, D., CAHEN, R., DING, H., AND JACQUEMIN, C. 2011. Sound level of detail in interactive audiographic 3D scenes. In *Procees of the International Computer Music Conference (ICMC)*.

THE WEB3D CONSORTIUM, 2008. X3D specification – part 1: Architecture and base components. http://www.web3d.org/x3d/specifications. Edition 2, ISO/IEC 19775-1.2:2008.

TSINGOS, N., FUNKHOUSER, T., NGAN, A., AND CARLBOM, I. 2001. Modeling acoustics in virtual environments using the uniform theory of diffraction. In *Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH, 545–552.

TSINGOS, N., GALLO, E., AND DRETTAKIS, G. 2004. Perceptual audio rendering of complex virtual environments. In *ACM SIGGRAPH 2004 Papers*, ACM, 249–258.

[3] http://www.topophonie.fr

Appendix B

Extended X3D representation – XML Schema

Representation of sound process

element : TPSoundProcess

```
1
2  <xs:element name="TPSoundProcess">
3      <xs:annotation>
4          <xs:appinfo>
5              <xs:attribute name="otherInterfaces" type="xs:string" ↩
                   fixed="X3DUrlObject"/>
6          </xs:appinfo>
7          <xs:documentation source="http://www.topophonie.fr"/>
8      </xs:annotation>
9      <xs:complexType mixed="false">
10         <xs:complexContent mixed="false">
11             <xs:extension base="X3DSoundSourceNode">
12                 <xs:sequence>
13                         <xs:element ref="TPGeometryProfileContainer"/↩
                               >
14                 </xs:sequence>
15                 <xs:attribute name="description" type="SFString"/>
16                 <xs:attribute name="url"         type="MFString"/>
17                 <xs:attribute name="model"       type="MFString"/>
18                 <xs:attribute name="activation" type="MFFloat"/>
19             </xs:extension>
20         </xs:complexContent>
21     </xs:complexType>
22 </xs:element>
```

In the above schema, we added four attributes (“description”, “url”, “model”, “activation”) among which there is no attribute “DEF”. It is because the “DEF” is a default attribute of sub–class X3DSoundSourceNode. Note that the child element TPGeometryProfileContainer is used to specify a 3D geometry object and an activation profile associated with the *sound process*. We can see the specification of element TPGeometryProfileContainer in Section 5.4.4.

Representation of sound source

element : TPSound

```
1
2  <xs:element name="TPSound">
3          <xs:annotation>
4                  <xs:appinfo/>
5                  <xs:documentation source="http://www.topophonie.fr"/>
6          </xs:annotation>
7          <xs:complexType mixed="false">
8                  <xs:complexContent mixed="false">
9                          <xs:extension base="X3DSoundNode">
10                                 <xs:sequence>
11                                         <xs:element ref="↩
                                               TPSoundProcess"/>
12                                 </xs:sequence>
13                                 <xs:attribute name="direction" type="↩
                                       SFVec3f " default="0 0 1"/>
14                                 <xs:attribute name="intensity" type="↩
                                       SFFloat " default="1"/>
15                                 <xs:attribute name="location" type=" ↩
                                      SFVec3f " default="0 0 0"/>
16                                 <xs:attribute name="maxBack" type=" ↩
                                      SFFloat " default="10"/>
17                                 <xs:attribute name="maxFront " type="↩
                                       SFFloat " default="10"/>
18                                 <xs:attribute name="minBack" type=" ↩
                                      SFFloat " default="1"/>
19                                 <xs:attribute name="minFront " type="↩
                                       SFFloat " default="1"/>
20                                 <xs:attribute name="priority" type=" ↩
                                      SFFloat " default="0"/>
21                                 <xs:attribute name="spatialize" type=↩
                                      "SFBool" default="true"/>
22                         </xs:extension>
23                 </xs:complexContent>
```

```
24          </xs:complexType>
25  </xs:element>
```

In the above schema, we added the same attributes of element Sound in the element TPSound, since the two elements have similar properties. The only difference is that TPSound has a child element TPSoundProcess, which we added in the sequence of indispensable child node.

Representation of profile functions in 1D, 2D and 3D

complexType :	TPProfileFunction
	TPProfileFunction1D
	TPProfileFunction2D
	TPProfileFunction3D
element :	TPProfileFunctionLinear
	TPProfileFunctionDelta
	TPProfileFunctionExscale
	TPProfileFunctionLinear2D
	TPProfileFunctionDelta2D
	TPProfileFunctionExscale2D
	TPProfileFunctionLinear3D
	TPProfileFunctionDelta3D
	TPProfileFunctionExscale3D

```
1
2   <xs:complexType name="TPProfileFunction" abstract="true" mixed="false↩
        ">
3           <xs:annotation>
4                   <xs:appinfo></xs:appinfo>
5                   <xs:documentation source="http://www.topophonie.fr"/>
6           </xs:annotation>
7           <xs:complexContent mixed="false"/>
8   </xs:complexType>
9
10
11  \begin{lstlisting}
12  <xs:complexType name="TPProfileFunction1D" abstract="true" mixed="↩
        false">
13          <xs:extension base="TPProfileFunction">
14          <xs:annotation>
15                  <xs:appinfo>
```

```
16                  <xs:attribute name="otherInterfaces" type="xs:string"↩
                        fixed="X3DUrlObject"/>
17                  </xs:appinfo>
18              <xs:documentation source="http://www.topophonie.fr"/>
19          </xs:annotation>
20              <xs:complexContent mixed="false">
21                 <xs:attribute name="mindist" type="SFFloat"/>
22                  <xs:attribute name="maxdist" type="SFFloat"/>
23                  <xs:attribute name="invert"  type="SFBool"/>
24              </xs:complexContent>
25      </xs:extension>
26  </xs:complexType>
27
28  <xs:element name="TPProfileFunctionLinear">
29      <xs:annotation>
30          <xs:appinfo>
31              <xs:attribute name="otherInterfaces" type="xs:string" ↩
                    fixed="X3DUrlObject"/>
32          </xs:appinfo>
33          <xs:documentation source="http://www.topophonie.fr"/>
34      </xs:annotation>
35      <xs:complexType base="TPProfileFunction1D"/>    <!-- no added ↩
            data -->
36  </xs:element>
37
38  <xs:element name="TPProfileFunctionDelta">
39      <xs:annotation>
40          <xs:appinfo>
41              <xs:attribute name="otherInterfaces" type="xs:string" ↩
                    fixed="X3DUrlObject"/>
42          </xs:appinfo>
43          <xs:documentation source="http://www.topophonie.fr"/>
44      </xs:annotation>
45      <xs:complexType base="TPProfileFunction1D">
46          <xs:complexContent mixed="false">
47              <xs:attribute name="middle" type="SFFloat"/>
48              <xs:attribute name="width"  type="SFFloat"/>
49          </xs:complexContent>
50      </xs:complexType>
51  </xs:element>
52
53  <xs:element name="TPProfileFunctionExscale">
54      <xs:annotation>
55          <xs:appinfo>
56              <xs:attribute name="otherInterfaces" type="xs:string" ↩
                    fixed="X3DUrlObject"/>
57          </xs:appinfo>
58          <xs:documentation source="http://www.topophonie.fr"/>
59      </xs:annotation>
60      <xs:complexType base="TPProfileFunction1D">
61          <xs:complexContent mixed="false">
```

```
62              <xs:attribute name="base" type="SFFloat"/>
63          </xs:complexContent>
64      </xs:complexType>
65  </xs:element>
66
67  <xs:complexType name="TPProfileFunction2D" abstract="true" mixed="↩
        false">
68          <xs:extension base="TPProfileFunction">
69          <xs:annotation>
70                  <xs:appinfo>
71                  <xs:attribute name="otherInterfaces" type="xs:string"↩
                        fixed="X3DUrlObject"/>
72                  </xs:appinfo>
73              <xs:documentation source="http://www.topophonie.fr"/>
74          </xs:annotation>
75              <xs:complexContent mixed="false">
76                 <xs:attribute name="revolution" type="SFFloat"/>
77                  <xs:attribute name="extrusion" type="SFFloat"/>
78                  <xs:attribute name="interpolation"  type="SFFloat"/>
79             </xs:complexContent>
80      </xs:extension>
81  </xs:complexType>
82
83  <xs:element name="TPProfileFunctionLinear2D">
84      <xs:annotation>
85          <xs:appinfo>
86              <xs:attribute name="otherInterfaces" type="xs:string" ↩
                    fixed="X3DUrlObject"/>
87          </xs:appinfo>
88          <xs:documentation source="http://www.topophonie.fr"/>
89      </xs:annotation>
90      <xs:complexType base="TPProfileFunction2D"/>    <!-- no added ↩
            data -->
91  </xs:element>
92
93  <xs:element name="TPProfileFunctionDelta2D">
94      <xs:annotation>
95          <xs:appinfo>
96              <xs:attribute name="otherInterfaces" type="xs:string" ↩
                    fixed="X3DUrlObject"/>
97          </xs:appinfo>
98          <xs:documentation source="http://www.topophonie.fr"/>
99      </xs:annotation>
100     <xs:complexType base="TPProfileFunction2D"/>
101 </xs:element>
102
103 <xs:element name="TPProfileFunctionExscale2D">
104     <xs:annotation>
105         <xs:appinfo>
106             <xs:attribute name="otherInterfaces" type="xs:string" ↩
                    fixed="X3DUrlObject"/>
```

```
107         </xs:appinfo>
108         <xs:documentation source="http://www.topophonie.fr"/>
109     </xs:annotation>
110     <xs:complexType base="TPProfileFunction2D"/>
111 </xs:element>
112 
113 <xs:complexType name="TPProfileFunction3D" abstract="true" mixed="↩
        false">
114         <xs:extension base="TPProfileFunction">
115         <xs:annotation>
116                 <xs:appinfo>
117                 <xs:attribute name="otherInterfaces" type="xs:string"↩
                        fixed="X3DUrlObject"/>
118                 </xs:appinfo>
119             <xs:documentation source="http://www.topophonie.fr"/>
120         </xs:annotation>
121             <xs:complexContent mixed="false">
122                <xs:attribute name="revolution" type="SFFloat"/>
123                 <xs:attribute name="extrusion" type="SFFloat"/>
124                 <xs:attribute name="interpolation"  type="SFFloat"/>
125            </xs:complexContent>
126     </xs:extension>
127 </xs:complexType>
128 
129 <xs:element name="TPProfileFunctionLinear3D">
130     <xs:annotation>
131         <xs:appinfo>
132             <xs:attribute name="otherInterfaces" type="xs:string" ↩
                    fixed="X3DUrlObject"/>
133         </xs:appinfo>
134         <xs:documentation source="http://www.topophonie.fr"/>
135     </xs:annotation>
136     <xs:complexType base="TPProfileFunction3D"/>    <!-- no added ↩
        data -->
137 </xs:element>
138 
139 <xs:element name="TPProfileFunctionDelta3D">
140     <xs:annotation>
141         <xs:appinfo>
142             <xs:attribute name="otherInterfaces" type="xs:string" ↩
                    fixed="X3DUrlObject"/>
143         </xs:appinfo>
144         <xs:documentation source="http://www.topophonie.fr"/>
145     </xs:annotation>
146     <xs:complexType base="TPProfileFunction3D"/>
147 </xs:element>
148 
149 <xs:element name="TPProfileFunctionExscale3D">
150     <xs:annotation>
151         <xs:appinfo>
152             <xs:attribute name="otherInterfaces" type="xs:string" ↩
```

```
                fixed="X3DUrlObject"/>
153         </xs:appinfo>
154         <xs:documentation source="http://www.topophonie.fr"/>
155     </xs:annotation>
156     <xs:complexType base="TPProfileFunction3D"/>
157 </xs:element>
```

In the above schema, all the concrete profile functions are derived from 1D functions, but one can extend at any time new 2D profile functions.

element : TPGeometryProfileContainer

group : TPProfileFunctionModel
TPProfileFunction1DModel
TPProfileFunction2DModel
TPProfileFunction3DModel

```
1
2  <xs:element name="TPGeometryProfileContainer">
3          <xs:annotation>
4                  <xs:appinfo>
5                          <xs:attribute name="otherInterfaces" type="xs↩
                               :string" fixed="X3DBoundedObject"/>
6                  </xs:appinfo>
7                  <xs:documentation source="http://www.topophonie.fr"/>
8          </xs:annotation>
9          <xs:complexType mixed="false">
10                 <xs:complexContent mixed="false">
11                         <xs:extension base="X3DChildNode ">
12                                 <xs:sequence>
13                                         <xs:group ref="↩
                                               ShapeChildContentModel" ↩
                                               minOccurs="0"/>
14                                         <xs:group ref="↩
                                               TPProfileFunctionModel" ↩
                                               minOccurs="0">
15                                 </xs:sequence>
16                                 <xs:attribute name="bboxCenter" type=↩
                                       "SFVec3f" default="0 0 0"/>
17                                 <xs:attribute name="bboxSize" type="↩
                                       BoundingBoxSize" default="-1 -1 -1↩
                                       "/>
18                         </xs:extension>
19                 </xs:complexContent>
20         </xs:complexType>
21 </xs:element>
22
```

```
23 <xs:group name="TPProfileFunctionModel">
24         <xs:annotation>
25                 <xs:appinfo>TPProfileFunctionModelis the child node ↩
                       function model corresponding to TPProfileFunction<↩
                       /xs:appinfo>
26                 <xs:documentation source="http://www.topophonie.fr"/>
27         </xs:annotation>
28         <xs:choice>
29                 <xs:group ref="TPProfileFunction1DModel"/>
30                 <xs:group ref="TPProfileFunction2DModel"/>
31                 <xs:group ref="TPProfileFunction3DModel"/>
32         </xs:choice>
33 </xs:group>
34
35 <xs:group name="TPProfileFunction1DModel">
36         <xs:annotation>
37                 <xs:appinfo>TPProfileFunction1DModel is the child ↩
                       node function model corresponding to ↩
                       TPProfileFunction1D</xs:appinfo>
38                 <xs:documentat ion source="http://www.topophonie.fr"/↩
                       >
39         </xs:annotation>
40         <xs:choice>
41                 <xs:element ref="ProfileFunctionLinear"/>
42                 <xs:element ref="ProfileFunctionDelta"/>
43                 <xs:element ref="ProfileFunctionExscale"/>
44         </xs:choice>
45 </xs:group>
46
47 <xs:group name="TPProfileFunction2DModel">
48         <xs:annotation>
49                 <xs:appinfo>TPProfileFunction2DModel is the child ↩
                       node function model corresponding to ↩
                       TPProfileFunction2D</xs:appinfo>
50                 <xs:documentat ion source="http://www.topophonie.fr"/↩
                       >
51         </xs:annotation>
52         <xs:choice>
53                 <xs:element ref="ProfileFunctionLinear2D"/>
54                 <xs:element ref="ProfileFunctionDelta2D"/>
55                 <xs:element ref="ProfileFunctionExscale2D"/>
56         </xs:choice>
57 </xs:group>
58
59 <xs:group name="TPProfileFunction3DModel">
60         <xs:annotation>
61                 <xs:appinfo>TPProfileFunction3DModel is the child ↩
                       node function model corresponding to ↩
                       TPProfileFunction3D</xs:appinfo>
62                 <xs:documentation source="\textbf{http://www.↩
                       topophonie.fr"/>
```

```
63          </xs:annotation>
64          <xs:choice>
65                  <xs:element ref="ProfileFunctionLinear3D"/>
66                  <xs:element ref="ProfileFunctionDelta3D"/>
67                  <xs:element ref="ProfileFunctionExscale3D"/>
68          </xs:choice>
69  </xs:group>
```

The element TPGeometryProfileContainer contains a sequence of indispensable child nodes to associate a *sound process* with a geometry object and an activation profile, as mentioned previously. However, the profile functions are enormous, and it is not convenient to list all of them in the element TPGeometryProfileContainer. So, the best way is to use the group node for grouping all possible child nodes in the models. Note that we can always extend these group nodes (TPProfileFunction3DModel, TP-ProfileFunction3DModel, TPProfileFunction3DModel) by new concrete profile functions.

Glossary

- LOD : Level of Detail
- $GLOD$: Graphics LOD
- $SLOD$: Sound LOD
- JND : Just Noticeable Difference
- $CBCS$: Corpus-based Concatenative Sound
- HCI : Human Computer Interactions
- VE : Virtual Environment
- $BRDF$: Bidirectional Reflectance Distribution Function
- LCJ : Law of Comparative Judgment
- HVS : Human Visual System
- API : Application Programming Interface
- OSC : Open Sound Control
- IAE : Interactive Audio Engine
- $L-system$: Lindenmayer system
- $W3C$: World Wide Web Consortium
- XSD : XML Schema Definition
- ANR : French National Research Agency
- DTD : Document Type Definition

- *IAEOU* : IAE Object for Unity