LISTEN LISBOA: Scripting Languages for Interactive Musical Installations

Cecile Le Prado, Stéphane Natkin

CEDRIC/ CNAM, Paris, France, cecile.leprado@free.fr,, natkin@cnam.fr

***Abstract---* This paper starts from an experimental interactive musical installation designed at IRCAM: "LISTEN LISBOA". This installation relies on a perceptual paradox: spectators are walking into a real space, see this space and at the same time hears through headphones a virtual sound space, mapped to the real one. In the first part of this paper we present this installation and more precisely the interactive musical "scenario" designed by the composer. We derive from this example general need for tools, used by composers working on this kind of installation. We show that these needs are close to those of video game level designers and associated scene languages with some extensions.**

Index Term – Sound Installation, Video Games, Scripting Languages, Space.

I. Introduction

During the last decade numerous augmented reality systems based on the mapping of a real space into a virtual sound space have been proposed. Spectators walk into a real space such as a factory, a museum or any kind of outdoor or indoor place, see this space at the same time hear, through headphones, a virtual sound space. This kind of system has several applications such as art installations, personal guided visits, audio help to drivers in a reduced visibility area, audio help in the maintenance of industrial plants. The virtual and the real space can have any shape and physical characteristics. The only basic constraint is that the two spaces must be homeomorphous, i.e. each feasible trajectory of a spectator in the real space has a corresponding continuous trajectory in the virtual sound space. Depending on the application other constraints may be added, such as the realistic neutral or abstract nature of the virtual acoustic space. The virtual space may be "stationary" or may depend on deterministic or random past events. For example a spectator can leave a "sound message" or a "sound trace" at a given place, which will be heard later by another spectator at the same place.

According to the size of the space, the audience and the quality of sound needed, this type of systems open numerous technical issues. The technical principles of this type of installation are discussed in [6] and [7], the implementation used in the LISTEN LISBOA project was designed and implemented as a part of the European project LISTEN [2]

This paper focuses on the composition language and interface that must be provided to a composer or to a sound designer in order to build this type of sound creation. In the first section we present in detail the LISTEN LISBOA experiments. In the second section we analyse how the needs must be addressed in a more general sound authoring tools. We show that these needs are closely related to the ones of classical level editing for video games and new types of applications like geo-localized games. We analyze also the relation with 3D scene description language like MPEG4 and VRML.

II. LISTEN LISBOA

A. History of LISTEN LISBOA

The LISTEN LISBOA experiment is derived from two other projects: SECRET LISBOA and LISTEN. SECRET LISBOA is an artistic radio broadcast composition. It is a commission from Radio France with the collaboration of IRCAM (Institut de Recherche et Coordination Acoustique/Musique). The composition consists of a visit in the city of Lisbon through its literature, its actual sounds, music compositions and interviews with artists. Sound materials use both, Soundfield 3D microphone recordings [12], and conventional stereo and mono recordings. The piece was composed and mixed using the Spat~ IRCAM environment [14]. The same piece was both broadcasted using a Dolby surround format [13] and recorded as a 5.1 musical composition. The piece was first presented in IRCAM, Paris 2005 and then in Hollywood for the ACE meeting in 2006.

LISTEN is a CEE research project whose goal was to provide a real time interactive sound space relating a real visual environment to an individual sound experience. LISTEN LISBOA is, from an audience's point of view, an individualized visit in the world of SECRET LISBOA. According to his/her physical displacement, the time and attention he/she pays to the different sound events, the listener experiences a personal listening of Lisbon landscape and memory. LISTEN LISBOA is, from the composer point of view, also an artistic sound experiment about interactive narration. The possible listener navigation, focus and memory are described through a script, which, combined to the sound and music material, is the essence of the composition.

Fig 1: Recording in Lisbon,

All the sound elements used in LISTEN LISBOA are derived from recorded sound in Lisbon, either monophonic, stereophonic or ambisonic (i.e. 3D sounds)(Figure 1). These elements have been transformed and edited using Protools and then are used in the interactive composition as sound objects for a binaural listening through headphones. More precisely the composition uses five sound streams: three mono recorded voices (Cecile, the Philosopher and the Actor) and two ambisonic sound spaces of Lisbon: the Harbor and the Musical Zone). These sound spaces are played in loop.

The visitor navigation is designed like in a video game using increasing level of complexity. In this experimentation, we define three levels as a sequence of tutorials. In the first level (Level 1) the listener has nothing particular to do except to walk in a virtual acoustic zone. He is always escorted by a mono sound source (voice) facing him. The goal is to invite the visitor to experiment a binaural listening and to understand the relation between the virtual soundscape and the real area of the installation. The second level (Act 2) allows to discover immersion in ambisonic sound and to explore two virtual zones mapped on the floor. The goal is to understand the invisible sound cartography. In the level3 (Act3.1 and Act 3.2) we introduce new type of constraints. There are related to the amount of time the visitor spends in a given area and also the ability to take care of a virtual character in the scene. According to these constraints, some new sound objects appear in the virtual sound scene. The difficulty increases and the listener need more attention to understand the semantic of voices but also the morphologic content of the sounds. This level is separated in two sub-acts that brings either to a shorter or to a more complex visit of the city.

The physical space is an empty small area (30 square meters). Only one user may be in this space at a given time. The user enters the space by a given points. He wears a position tracker that is able to detect his position and the direction of his head (Figure 3). So the system acquires five parameters: the three coordinates of the spectator location and two angles that define the head orientation. It is the only inputs send to the system, i.e.; the spectator interaction is controlled only by its traveling in the real space and the movements of his head. The Figure 2 summarizes the system.

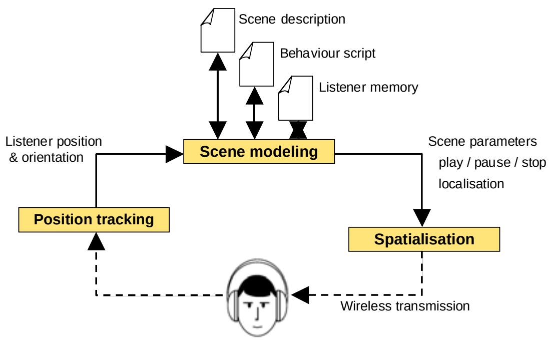

Fig 2: The LISTEN LISBOA system

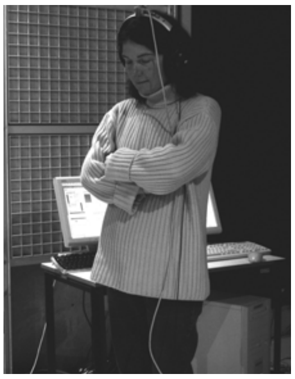

Fig 3: The listener, headphones and head tracker

The piece has the form of an interactive musical narration that can be described by the informal script illustrated in

Figure 4 and 5. In this script the “When” instruction has to be considered as a guarded command.

B. *Analysis of LISTEN LISBOA script*

LISTEN LISBOA was implemented under the Max MSP software by the Room Acoustic Research Laboratory of IRCAM. Over the MAX software, LISTEN provides an environment able to control the system described by Figure 6. During the LISTEN LISBOA project a geometrical “design” and a “control” [2] (Figure 7) interface was designed by IRCAM.

The design interface is directly related to the application of LISTEN (dynamical visit of a museum), where the real space and the virtual spaces are tightly mapped. It is a 2D “level design” environment (see the next section). The objects controlled in the scene are the listener, the sound sources and virtual zones to control the virtual acoustic. This interface was not available for the design of LISTEN LISBOA, but it has the basic geometrical properties needed for this type of application. The other important point is the scripting language. The script capabilities of LISTEN SPACE are those of the Max MSP software, [11]. Max MSP provides both a graphic programming language (related to synchronous languages) and an object oriented declarative language. The scripting language is not specification oriented and may be too complex for a direct use by a composer.

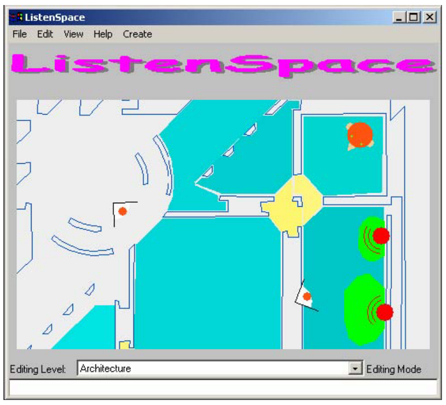

Fig 6: The LISTEN design Interface [2],

Fig 7: LISTEN LISBOA control interface,

The analysis of LISTEN LISBOA script shows the following properties:

- It is event and space oriented : the when clause defines the reaction to an event produced by the position of the listener in the space.
- It relies on a scene in the meaning of scene description languages (VRML or MPEG 4 for example) see Figure 8.
- It is procedural (or narrative): it defines a finite non deterministic order of events.
- It manipulates objects in the programming meaning of this term.
- It use both synchronous (audio streams) and asynchronous events (entering in a zone).
- All the asynchronous events are collision detections between the auditor (its avatar) and virtual objects
- Sound real time operators used are the real time mixing of streams, sound localisation and reverberation.

According to the possible evolutions of this work a specification environment should also includes the ability to manage other inputs (messages lefts by the listener though a microphone in a given place), a complex memory of the events that occurred during a visitor experiment, a more object oriented specification (for non narrative environment) and the ability to use sophisticated sound functions (real time synthesis and transformation), taking Max MSP functionalities as a reference.

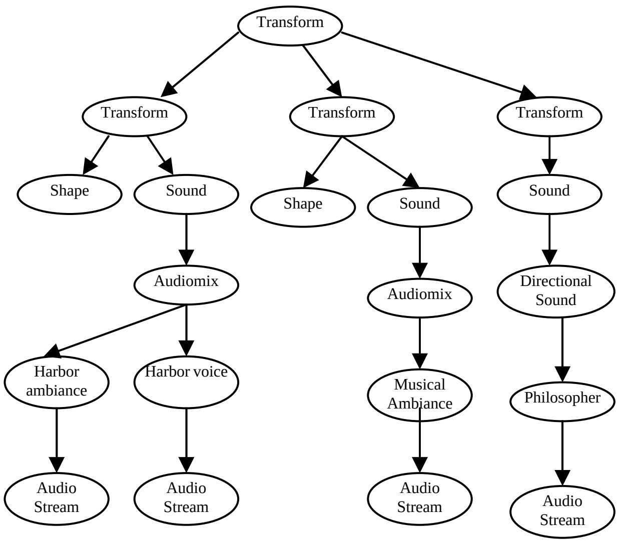

Fig 8: a scene representation of LISTEN LISBOA

```
Act 1
When the presence of an listener at the initial location is detected
    Wait a while
    Start Cecile voice and make it audible to the listener in front of him,
    The voice follows the spectator until the end of Cecile sound stream
    Go to Act2
Act 2
    Create two zones, Harbour and Musical, in the virtual sound space, mapped to the real space according to the Figure 3. Start the Harbour and The Musical streams
    Wait until the spectator reaches a zone
    When the Listener is in a zone
        Make the corresponding stream audible. The level of sound depends on the distance from the centre of the zone to the listener.
    When the Listener stays in the Musical zone more than one minute
        Go to Act 3.1
    When the Listener stays in the Harbour more than two minutes
        Destroy the Musical zone and stop the corresponding stream.
        Go to Act 3.2
    When the Listener is moving from one zone to the other
        Wait two minutes
        Go to Act 3.1
Act 3
Act 3.1
    When the Listener starts Act3 in the Musical zone or enters in this zone for the first time and stays a while in the middle of the Musical zone
        Locate the philosopher at a fixed
            point of the real space, and define a cone of audibility (see Figure 3), Start and Pause the stream
    When the Listener starts Act3 in the Harbour zone or enter in this zone for the first time and stays a while in the middle of the Harbour zone and the Philosopher stream has reach its end
        Start and Play the Actor stream audible behind the listener, the voice follow the listener.
    When the Listener head orientation enters the audibility cone space
        Play the Philosopher stream
    When the Listener head orientation exits the audibility cone space
        Pause the Philosopher stream
    When the listener is in a zone
        Make the corresponding stream audible. The level of sound depends on the distance from the centre of the zone to the listener.

    When the actor stream reaches a given point (A specified word)
        Increase progressively the reverberation levels in Harbour and Musical zones
        End
Act 3.2
    When the Listener enters the Harbour zone for the first time
        Start and play the actor stream audible behind the listener, the voice follow the listener.
    When the Listener is in the Harbour zone
        Make the corresponding stream audible. The level of sound depends on the distance from the centre of the zone to the listener.
    When the actor stream reaches a given point (A given word in his talk)
        Increase progressively the reverberation levels in Harbour zone
        End
```

Fig 4: LISTEN LISBOA script

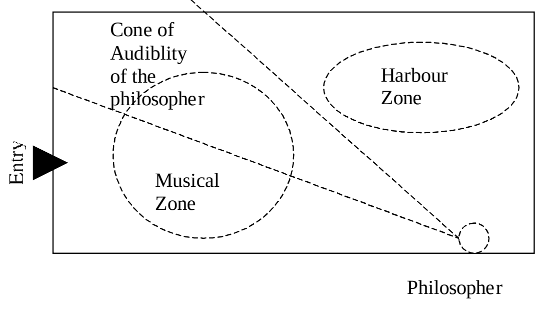

Fig 5: Mapping of virtual fixed objects (dotted) in the real space

III. Level design in video Games and scene description languages

A. *Level Design*

From a macroscopic point of view, the main structuring aspects of a video game are the map of the universe and the game levels. A level can be defined as the main linear part of the game structure. Each level is associated with a main subgoal of the game goal. The possible ordering of levels is limited by the logical structure of the game puzzle. Each level has a narrative or perceptual necessity [1]. Levels are also generally associated with a piece of the map.

The level design is the main step where the scenario of a game takes place. It induces a partially ordered set of actions that the player must perform to end the level, defines the goals assigned to the player and limits the number of possible effective actions of the player. Level design is the only constructive way to simulate, in a game, a classical narrative construction scheme. But it can not be based on the time driven presentation of media, playing with the memory and the emotion of the spectator through passive perceptions. It must use the mix of immersion factors and rely not on time but on space and logic constructions. A level of the game is a mix of a virtual space, a set of puzzle to be solved in this space and the main actions to be done by the player to reach a given goal. The level is first defined by the geometry of the space: a given maze, a race circuit. Then the level designer chooses the positions and actions associated with the objects in this level, defining an interactive narration scheme through all the possible paths. To keep the sensation of freedom, several solutions are used: first, a set of independent actions can be performed in any order, in more complex games the player can pursue, in the same space, several goals in parallel. These principles of writing have been analyzed and formalized [4].

A level editor relies on a 3D representation of the Level space. Figure 9 is the interface of the Half Life 2 (HL2) Level Editor [1]. The level designer first works on the geometry, building corridors, walls, doors, mountains and rocks, to control the player avatar displacements. Then he places in the space some objects (enemies, keys for the doors, weapons, ammunition, life potions) which define the possible actions of the player. The places of these objects are related to the quest logic: you must be able to find a key before opening the door and a weapon before killing a monster. At last the level designer places triggers to initiate some script execution. Generally triggers are related to collisions detection. For example, to initiate an attack against the player avatar at a given point an invisible barrier is created. The crossing of this barrier triggers the attack script.

[1] Half Life 2 is a famous First Person Shooter game released by Valve in 2005.

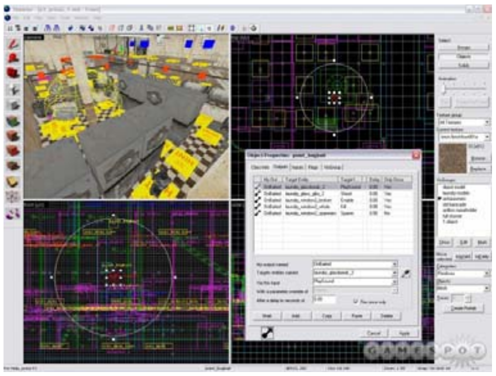
Fig 9: HL2 level Editor,

Messages	**Description**
Touched	An object or surface was touched by another object. References to both collision participants can be retrieved.
Entered	For sectors, called each time a new object enters the sector
Damaged	Called whenever the object would take damage from weapons or explosions. References to the cause of the damage and the type of damage are provided to the handler.
Created	Called on a new object when it is first created
Killed	Called when the object is about to be removed from the game
Crossed	Called for an adjoin plane whenever an object crosses it
Arrived	Called when a moving object reaches its destination
Timer	A timer event set by the script has expired
Sighted	An object is seen by the player for the first time

Fig 10: Events in the COG scripting language

Level editors are more or less narrative oriented: either most of the script is associated with the global ordering of events (like in LISTEN LISBOA) or is related to the behaviour of autonomous objects receiving and sending messages to other objects. The editor may be also more or less synchronous oriented. A synchronous oriented environment is designed from an “interactive movie” point of view: events trigger the execution of recorded sequences of synchronous media that can be partially ordered (like, for example, in the web specification language SMIL). Asynchronous environment triggers mainly the execution of real time rendered media in response to events. The tools to develop games like Indigo Prophecy/Fahrenheit[2] have of this narrative and synchronous point of view. Renderware [15], the tool used to develop games like the GTA[3] series, is object and asynchronous oriented.

Figure 10 shows the events used in the game scripting language COG [16]. COG was used to develop the Jedi Knight game. The type of events listed are closed to the one used in the LISTEN LISBOA script, in a different context (replace for example killed by destroyed). We can find the same types of

[2] Indigo Prophecy (US name)/ Farenheit (EU name) is a game developed by Quantic Dreams published by Atari and launched in 2005

[3] GTA (Grand Theft Auto) is a series of famous games developed by Rockstar studio

objects in the script of the Figure 11 (event triggered by "when shot", collision triggered by "GetWallCel(sign) == 0" and mixing of sound streams : "sound exp_sound desc=played"). In counterpart the scripting language is too close to a programming language like C++ and is still too complex for non technical users.

```
#00_neosign.cog
#
#if this cog will cycle through frames 0-(lastFrame-1), at framerate fps
#if damaged, it will go to frame lastFrame and stop, create sparks and sound
symbols
message startup
message damaged

surface sign mask=0x448
float fps=2.0 desc=speed of anim
template sparks=+sparks desc=created when shot
sound exp_sound desc=played when shot
end

code
startup:
// Start the animation looping but skipping the first 2 frames
SurfaceAnim(sign, fps, 0x5);
return;

damaged:
if (GetWallCel(sign) == 0)
return; StopSurfaceAnim(sign);

if (exp_sound)
PlaySoundPos(exp_sound, SurfaceCenter(sign), 1.0, -1, -1, 0);

SetWallCel(sign, 0); CreateThing(sparks, GetSourceRef()); return;
end
```

Fig 11: A COG script [16]

As a conclusion, it seems that level editors provide all the tools to specify a sound installation, like LISTEN LISBOA, with three exceptions:

- The language should be as simple as possible
- The virtual space is not related to a real space, there is the need of a map between the two worlds (Natkin 2004)
- The sound operators (both in the space interface and the script) are restricted

B. *Scene Description Languages*

Scene description languages like VRML and MPEG 4, allows describing the relation between virtual objects in a space, in particular from a sound point of view.

"The systems part of the MPEG-4 addresses the description of the relationship between the audio-visual components that constitute a scene. The relationship is described at two main levels.

- The Binary Format for Scenes (BIFS) describes the spatio-temporal arrangements of the objects in the scene… The scene description provides a rich set of nodes for 2-D and 3-D composition operators and graphics primitives.
- At a lower level, Object Descriptors (ODs) define the relationship between the Elementary Streams pertinent to each object (e.g. the audio and the video stream of a participant to a videoconference)…" [5]

The audio description of an object may allows mixing recorded and synthesized sounds, to control in detail spatialization and more generally real time effects applied to sound object.

"MPEG-4 coding of audio objects provides tools for both representing natural sounds (such as speech and music) and for synthesizing sounds based on structured descriptions. The representation for synthesized sound can be derived from text data or so-called instrument descriptions and by coding parameters to provide effects, such as reverberation and spatialization. The representations provide compression and other functionalities, such as scalability and effects processing." [5].

The figure 12 illustrates the control of fade in and out according to the location of the user avatar.

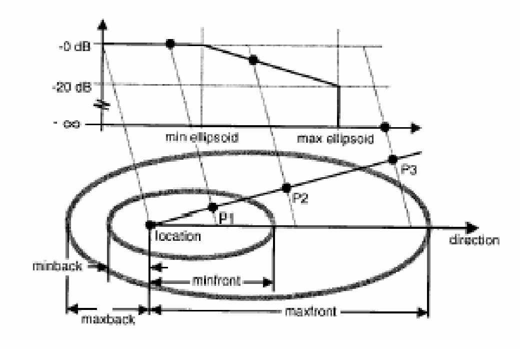

Fig 12: A sound diffusion region in VRML

In counterpart the formal representation of a scene relies on computer languages, that are a poor and complex interface for an artist. Thus an audiovisual tool must be built in top of it.

IV. CONCLUSION

In this paper, starting from an art sound installation, we have shown the need of a new type of space oriented sound specification tools. This tool is clearly related to level editors from its interface and to scene language in terms of sound functionalities. It must be able to cope with some complex relations between a virtual world and a real world. The same type of needs appears in the development of geo-localized games like Botfighter 2, where the evolution and the mapping of a real space in a virtual space are triggered by player movements [9].

We can raise three types of needs:

- The ability to construct complex 3D objects including complex sound functions in a scene. MPEG4 provides a good framework for this needs:

- The ability to edit audio streams using natural and synthetic audio and effects. A real time sound mixer like ISACT of creative labs or XSACT of Microsoft [4] provides such a function.
- The ability to edit in a simple and creative way complex 3D scene able to map real and virtual objects. Existing game level editors provide a good framework for this point.

We have started to work on a new type of tool, connecting this three types of tools and defining a mapping between real and virtual objects [4], but the point of view of numerous sound designers and composers writing interactive music is needed to design such a tool efficiently and we are working to get this information through a web inquiry.

ACKNOWLEDGMENT

We would like to thank all the people who have worked on SECRET and LISTEN LISBOA, in particular Michel Creïs, Olivier Delerue, Romain Kronenberg, Gilles Mardirossian, Emmanuel Rio and Olivier Warusfel . We would like also to thank Professor Gerry Kim for his deep work on the first version of this paper.
.

REFERENCES

[1] B. Bates, 2000. Game Design: “ The Art & Business of Creating Games“, Prima Tech Ed.W. -K. Chen, *Linear Networks and Systems* (Book style). Belmont, CA: Wadsworth, 1993, pp. 123–135.
[2] O. Delerue, O.Warusfel, “Authoring of Virtual Sound Scenes in the Context of the LISTEN Project“, AES 22, Helsinki, 2002.
[3] G. Eckel, "Immersive Audio-Augmented Environments", Proceedings of the 8th Biennial Symposium on Arts and Technology at Connecticut College, New London, CT, USA, 2001.
[4] M. Emerit, C. Le Prado, S. Natkin, Game Audio Writing, “A game audio technology overview“ Sound and music computing Conference SMC06, Marseille, 2006.
[5] S. Grunvogel, S. Natkin, L. Vega, “A New Methodology for Spatiotemporal Game Design“, CGAIDE 2004. Reading, UK, Nov 2004C.
[6] R. Koenen (ed), Overview of MPEG 4 standard, ISO/IEC JTC1/SC29/WG11 N4668, March 2002.
[7] S. Natkin , “Mapping a Virtual Sound Space into a Real Visual Space“, ICMC, Berlin, 2000.
[8] S. Natkin, F.Schaeffer, “ A Distributed and Mobile Architecture for the Mapping a Virtual Sound Space into a Real Visual Space“ ICMC, Goteborg, 2001
[9] S.Natkin, C. Yan, “Analysis of Correspondences between Real and Virtual Worlds in General Public Applications“. In CGIV05 Computer Graphics, Imaging and Visualization, Beijing, 25-28 July 2005, IEEE, 2005.
[10] S.Natkin, “Video Games and Interactive Media: A Glimpse at New Media Entertainment“, A.K Peters, Wellesley, MA, 2006
[11] Max MSP tutorials 2006 http://www.cycling74.com/section/tutorials
[12] Soundfield web site www.soundfield.com
[13] Dolby Web site www.dolby.com
[14] J.M. Jot, A Real-Time Spatial Sound Processor for Music and Virtual Reality Applications, ICMC, Banf, Sept 1995
[15] Site of Renderware, www.renderware.com
[16] R. Huebner, Adding Language to game engine, Game Developer Magazine, Sept 1997 and http://www.gamasutra.com/features/19971003/huebner_01.htm