Chapitre 6. L'OUÏE OU L'IMMERSION SONORE

Le son est une partie intégrante du monde réel. Il est riche en information et provient de plusieurs endroits : il est donc tridimensionnel.

Le son 3D est la clé pour produire un environnement de réalité virtuelle réaliste. Il joue différents rôles dans les environnements virtuels produisant ainsi :

- **Des informations complémentaires :** Les échos et les réflexions du son donnent des informations sur l'orientation des objets, leurs directions, et leurs distances, ainsi que les dimensions des environnements (les résonnances dans un grand amphithéâtre sont différents que celles dans un petit bureau). Les sons permettent aussi une bonne immersion dans l'espace : Ceux venus de l'extérieur du champ visuel décrivent des espaces non vus. Ils décrivent aussi les propriétés des objets dans le monde virtuel : Les sons diffèrent selon les natures des surfaces, et les caractéristiques physiques des objets.
- **L'interactivité de l'interface** : Les sons constituent une partie intégrante de tout système d'interfaçage homme-machine. Ils peuvent être des réponses à des commandes ou des alertes et avertissements du système ou de la scène virtuelle.
- **Modalité d'interaction alternative :** Les sons peuvent représenter également des canaux de représentation à part entière utilisant la reconnaissance vocale, et la synthèse des sons.

1.1. Enregistrement et reproduction des sons 3D

Ils existent trois méthodes pour acquérir le son en vue de le reproduire dans les scènes virtuelles. Ces méthodes diffèrent de par leurs moyens et protocoles d'enregistrement :

1. **Enregistrement monaural :** En utilisant un seul microphone, sans aucune information sur la position du son.
2. **Enregistrement Stéréo :** utilisant deux microphones distants. L'information contient la position du son (direction et source qui sont généralement perçus dans la tête de l'auditeur !).
3. **Enregistrement binaural :** Cet enregistrement semble plus réel. Les microphones simulant les propriétés de capture des oreilles sont placés sur une tête imaginaire. Les sons sont filtrés par des HRTF[1] pour générer des sons réalistes.

1.2. Synthèse des sons 3D

La synthèse consiste à traiter le signal sonore pour produire des sons réalistes suivant les positions des sources, les dimensions des environnements, ainsi que les autres caractéristiques physiques. Cette synthèse doit respecter les contraintes

[1] La fonction de transfert relative à la tête ou HRTF (Head-Related Transfer Function).

techniques liées à l'immersion comme le rendu en temps réel, la fidélité à l'amplitude et à la nature des sons, et le respect des propriétés physiques du temps et de l'espace.

a. Le rendu sonore :

Le rendu sonore est souvent utilisé pour accompagner les animations. Les sons généralement sont associés aux différents objets. Ces sons sont soit échantillonnés, soit générés artificiellement. Ce rendu considéré comme un pipeline à quatre niveaux :

1. Génération de chaque son associé à un objet séparément des contraintes de l'environnement ou de l'espace.
2. Association des sons aux mouvements des objets en tenant compte des distances et des propriétés de l'environnement.
3. Calcul des convolutions nécessaire à la description des sources sonores dans la scène.
4. Association des convolutions aux différentes sources sonores pour produire la scène sonore finale.

La technique exploite la similarité entre la lumière et le son pour calculer les convolutions. La source propage les ondes sonores dans toutes les directions. Ces ondes sont reflétées et réfléchies et réfractées suivant l'environnement acoustique. Les ondes interagissent avec plusieurs objets dans la scène. Le son final est l'intégral de tous les signaux associés aux différents trajets entre la source et l'auditeur. La contrainte majeure qui reste un défi à relever pour tout concepteur est la limite de calcul en temps réel, notamment pour les scènes les plus complexes.

b. Approximation 3D : Cette approximation consiste vise à augmenter la perception de l'immersion, elle est résumée en quatre étapes :

1. Récolte des informations sur le trajet Source – Utilisateur (Milieu, obstacles).
2. Traiter les sons émis par des algorithmes simples, ou des effets sonores classiques.
3. Les écouteurs sont utilisés pour jouer le son final suivant la position de l'utilisateur en offrant le rendu en stéréo.
4. Positionnement des haut-parleurs : Dans les cas des systèmes de réalité virtuelle à faible coût.

1.3. Les systèmes sonores pour la RV

Un système sonore pour la réalité virtuelle est un pipeline à différents niveaux satisfaisant certains critères dont nous citons les plus cruciales :

a. **Localisation 3D :** Calcul les positions des sources des sons virtuels correspondant aux objets virtuels associés.
b. **Simulation acoustique :** Pour percevoir les propriétés spatiales du monde virtuel comme les dimensions de la salle, et les propriétés de réflexion des murs.
c. **Temps de traitement :** Le compromis à gérer entre le réalisme des sons (tenant en compte les propriétés physique des matériaux), et le traitement en temps réel est très difficile à gérer.

1.4. Les périphériques du son

L'implantation matérielle des sons peut être classifiée comme suit :

a. Playback :

- **Haut-parleurs :** mieux adaptés à CAVE, moins cher, mais sensibles aux réflexions et aux obstacles.
- **Ecouteur (oreillettes) :** Contrôle spatial (deux sons différents), et le son est direct (pas de réflexion).

b. Haut-parleurs multiples : pour simuler l'ouïe naturelle : plusieurs sons, de différentes intensités, pour générer des sons « naturels ».

1.5. La fonction de transfert relative à la tête (*Head-Related Transfer Function HRTF*)

Il s'agit ici d'un filtre capable de caractériser mathématiquement la position de la source sonore suivant la position de la tête de l'utilisateur et les transformations apportées aux sons par sa tête, ses pavillons, et ses conduits auditifs.

Pratiquement, les sons sont capturés par des petits microphones placés sur les oreilles. Ces sons sont comparés avec les sons originaux (des sources sonores) pour calculer les filtres HRTF associés.

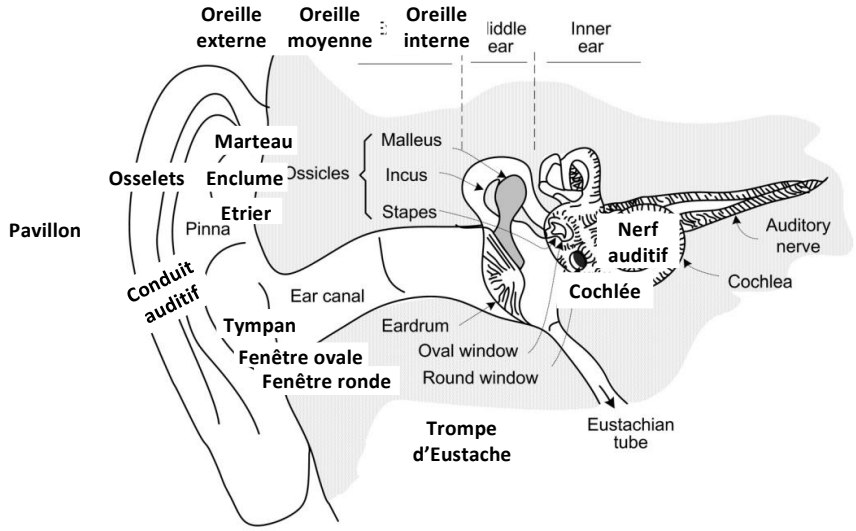

Figure 1 Section sagittale du système auditif périphérique humain

Principe de la fonction :

Le son dans un espace libre atteint les oreilles après des interactions avec la structure anatomique : Il est diffracté et réfléchi par le torse, la tête, le pavillon et les conduits auditifs (voir Figure 1). Le son binaural résultant contient une variété de types d'informations de localisation[2] telle que la *ITD*, la *ILD* et l'information spectrale :

a. *ITD* : *Interaual Time Difference*, qui représente la différence temporale de perception entre les deux oreilles.

b. *ILD* : *Interaural Level Difference*, qui représente la capacité des neurones sensitives à différencier les stimuli des deux oreilles séparément. Cette propriété permet de mettre à l'ombre (en arrière-plan) les sons acpturés.

La fonction de transfert relative à la tête (HRTF) décrit les effets de filtrage imposés par la structure anatomique modérés par les données de localisation. De ce fait, la fonction de transfert est spécifique à chaque individu en réalité. La reproduction synthétique se base sur une modélisation générique tenant compte les capacités moyennes humaines.

[2] La localisation sonore consiste à l'habilité de l'écouteur à identifier la position ou l'origine du son perçu dans sa direction et distance.

Chapitre 7. NORMES ET FORMATS

Dans ce chapitre, nous allons présenter certains normes et formats électroniques liés à la réalité virtuelle. Comme la réalité virtuelle est omniprésente dans notre vie quotidienne et dans tous les produits technologiques de nos jours, chaque fournisseur présente un côté RV plus ou moins élaboré dans sa nomenclature.

2.1. La norme RV dans le WEB : VRML

Le VRML est un acronyme pour désigner *Virtual Reality Markup Langage.* C'est un langage (ou plutôt un script) pour décrire des scènes virtuelles bidimensionnelles, tridimensionnelles ou plus, avec ou sans du son, sur la machine.

Il est multiplateforme et permet la publication des scènes tridimensionnelles sur le *web* en définissant ainsi un canal de communication standard en matière de réalité virtuelle. Le VRML est :

- Simple à écrire et à utiliser.
- Optimal en termes de ressources mémoire et processeur.
- Modulaire : Des modules crées peuvent être réutilisés pour bâtir de nouveaux projets ou intégrer d'autres scènes.

Les fichiers *VRML* portent l'extension *wrl* (**.wrl* venant de *world*). Pour les visualiser nous avons besoin de visualiseurs à part ou de modules et extensions disponibles dans les explorateurs du web.

2.1.1. Description de scènes dans VRML

Les scènes virtuelles dans les scripts *VRML* sont décrites par une hiérarchie dans un fichier *wrl*. Les entités de base dans le script sont appelés nœuds (54 nœuds différents). Ces nœuds enregistrent leurs données dans des champs (20 champs différents). Ces champs peuvent être à valeurs scalaires ou vectorielles.

2.1.2. Structure d'un fichier *wrl*

Un fichier *wrl* est structuré comme suit :

- **Entête :** Pour préciser au navigateur la nature du fichier et la version (***1.0*** ou ***2.0***) ainsi que le jeu de caractère utilisé (***utf8*** par exemple).
- Commentaires (Optionnels) : précédés par ***#***
- **Nœuds :** Introduit par le type (Commençant par une lettre majuscule) suivi d'une paire d'accolades contenant la définition du nœud.

2.1.3. Les distances dans VRML

- Les distances : en mètre.
- Les angles : en radian.
- Le temps : en seconde.
- Les couleurs : sont exprimés dans le système R V B (rouge, vert, bleu).

2.1.4. Exemple de quelques champs

a. Le nœud shape

Ce nœud définit tous les objets (formes) visibles, il contient deux champs :

- ***appearance*** (Facultatif)
- et ***geometry*** (Obligatoire).

Le script suivant décrit un exemple simple d'utilisation du nœud ***shape*** :

```
01. #VRML V2.0 utf8
02. # example.wrl
03. Shape {
04.   appearance NULL
05.   geometry NULL
06. }
```

Le champ ***geometry*** peut prendre l'une de ces valeurs relatives aux différents objets manipulables en *VRML* :

```
01. Box
02. Cone
03. Cylinder
04. ElevationGrid
05. Extrusion
06. IndexedFaceSet
07. IndexedLineSet
08. PointSet
09. Sphere
10. Text
```

Le champ ***appearance*** spécifie les caractéristiques visuelles de l'objet à tracer. Le script suivant en est un exemple simple :

```
01. Appearance {
02. material NULL
03. texture NULL
04. textureTransform NULL
05. }
```

b. Le nœud *material*

Commençant par ce script qui illustre une scène simple :

```
#VRML V2.0 utf8
#sample vrml file
DEF cube Box { size 2 2 2 }
DEF cone_ Cone { bottomRadius 1 height 2 side TRUE bottom TRUE }
DEF cylindre Cylinder { radius 1 height 2 side TRUE bottom TRUE top TRUE }
DEF sphere_ Sphere { radius 1 }
Shape{
  geometry USE cube
  appearance Appearance {
   material Material {
      ambientIntensity 0.3
      diffuseColor 0.5 0.9 0.8
      specularColor 0 0 0
      emissiveColor 0 0 0
      shininess 0.9
      transparency 0
   }
  }
}
```

Ce nœud permet de définir le matériau. Les paramètres sont soit des composantes de couleurs en RVB normalisées entre 0 et 1, soit des coefficients également compris entre 0 et 1 :

- ***diffuseColor*** définit la couleur de l'objet, au sens le plus général du terme. Les lumières de la scène influent sur l'apparence de l'objet grâce à cette couleur, en fonction de l'angle de réflexion et de la lumière sur l'objet.
- ***ambientIntensity*** correspond à l'influence de la lumière ambiante de la scène sur l'objet. La lumière ambiante est aussi produite par les lumières de la scène, mais ne dépend pas des diverses réflexions (nous parlons de lumière isotrope).
- ***specularColor*** définit la couleur des rayons lumineux réfléchis sur la surface de l'objet et renvoyés vers le point de vue de l'utilisateur. C'est ce qui sert à produire les reflets localisés sur des boules métalliques par exemple.
- ***emmissiveColor*** définit une couleur propre à l'objet, comme s'il était luminescent. C'est à dire qu'en l'absence de lumière, l'objet émet cette couleur.
- Les champs ***shininess*** et ***transparency*** définissent les effets liés à l'ombre et à la transparence des objets.

c. La syntaxe de *DEF* et *USE* :

Il est possible de réutiliser des nœuds précédemment décrits grâce aux directives ***DEF*** et ***USE***.

La directive ***DEF*** permet d'assigner un identifiant (nom) à un nœud particulier. Par exemple, pour définir un «cube» de taille 3, nous pouvons utiliser la syntaxe suivante :

```
DEF cube Box { size 3 3 3 }
```

Ensuite, au lieu de réécrire le nœud et l'ensemble de ses paramètres à chaque fois, nous pouvons simplement écrire :

```
01. USE cube
```

L'appel à DEF exécute réellement le nœud en question. Il ne s'agit donc pas seulement d'une définition, car le nœud défini prend effet immédiatement.

d. Quelques exemples :

Dans ce qui suit, nous présentons quelques exemples que vous pouvez intérpreter sur des visualiseurs VRML afin de vous familiariser avec le langage :

Exemple 1 :

```
01. #VRML V2.0 utf8
02. # example1.wrl - a yellow box
03. Shape {
04.     geometry Box { }
05.     appearance Appearance {
06.         material Material {
07.             diffuseColor    1.0  1.0  0.0       # red, green, blue
08.         }
09.     }
10. }
```

Exemple 2 :

```
01. #VRML V2.0 utf8
02. # example1.wrl - a yellow box
03. Shape {
04.     appearance Appearance {
05.         material Material {
06.             diffuseColor 0.3 0 0.4
07.         }
08.     }
09.     geometry Cone {
10.         height 3.0
11.         # radius defaults to 1.0
12.     }
13. }
```

Exemple 3 :

```
#VRML V2.0 utf8
# example1.wrl - a yellow box
WorldInfo {
    title "yellow box" }
NavigationInfo {    type "EXAMINE"}
Group {
  children [
    DEF base Transform {
      translation 0 -1.5 0
      children [
          Shape {
              appearance Appearance {
                  material Material {
                      diffuseColor 0.8 0 0 } }
              geometry Cylinder {
                  height 0.75 } }
          DEF lowerArm Transform {
              center 0 -1.875 0
              translation 0 1.875 0
              children [
                  Shape {
                      appearance Appearance {
                          material Material {
                              diffuseColor 0 0.8 0 } }
                      geometry Box {
                          size 0.5 3 0.5 } }
                   DEF upperArm Transform {
                      center 0 0 -1
                      translation 0 1.75 0.75
                      children [
                          Shape {
                              appearance Appearance {
                                  material Material {
                                      diffuseColor 0 0 0.8
                                      } }
                                  geometry Box {
                                      size 0.5 0.5 2 } }
                          ] }
                  ] }
          ] }
  ]
}
```

2.2. Une nouvelle norme dans le WEB : x3D

Il s'agit ici d'un format de fichier graphique orienté 3D que le consortium Web3D a défini dans le but de succéder à VRML. La finalité reste la même que pour VRML, soit définir un format interopérable sur le WEB. Sa structuration est de type graphe de scènes exprimé soit :

- En syntaxe VRML classique ;
- En syntaxe XML ;
- En binaire.

Le fichier x3D (*e**x**tensible **3D** graphics*) définit un graphe de scènes qui n'est d'autre qu'un graphe acyclique direct décrivant un monde virtuel tridimensionnel incluant les objets 3D et leurs interactions. Il est donc similaire à VRML, avec quelques ajouts propres à lui.

Parmi ces ajouts, nous pouvons citer *h-anim* qui est un standard pour définir les spécifications interchangeables pour les figures humanoïdes. C'est un standard introduit pour servir de base dans les animations et les jeux vidéo interactifs avec acteurs.

2.3. La vidéo stéréoscopique (vidéo 3D)

Appelé aussi vidéo immersifs, il s'agit d'un format non spécifié commercialisé pour visualiser les vidéos utilisant les lunettes immersives (voir section **Error! Reference source not found.**). Le principe est très simple, la vidéo est diffusée en double image gauche et droite, les deux images sont presque identiques, décalées de quelques pixels. Lorsque l'utilisateur regarde la vidéo avec les stéréo-lunettes à une distance suffisamment proche, les lentilles procurent la perception tridimensionnelle de la scène.