> SoundDelta project

Web site: Sound Delta

Monday 10th march Meeting: Graphics work for audio rendering

Grid decomposition

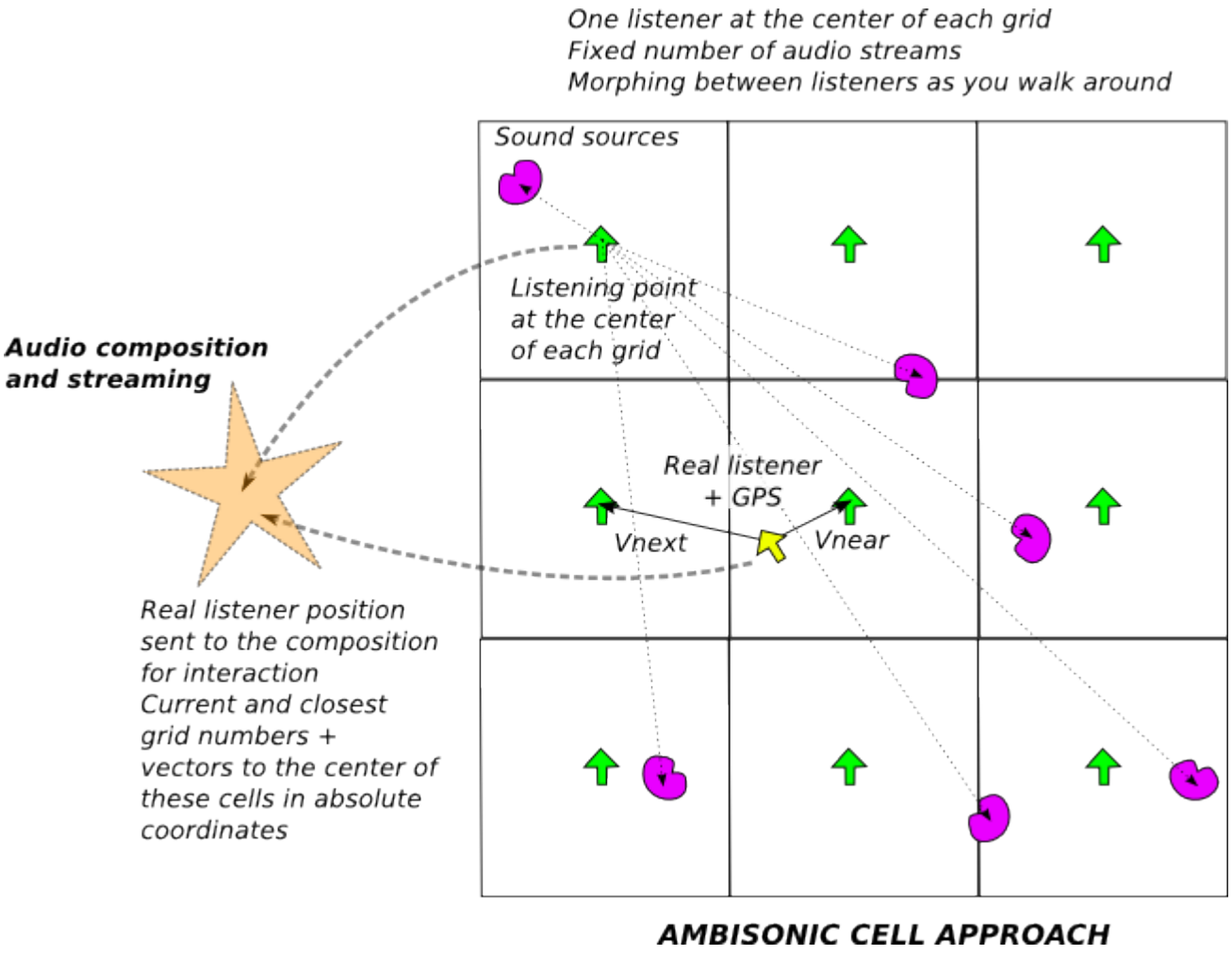
Scene + grid + viewpoint: occlusion would not work

To kinds of cells audio cells and wifi cells

You cannot make a blend between wifi channel

Each audio cell has a UDP port number: morphing audio channels coming from different ports

The PAD receives two streams: your stream + the stream of the neighboring cell



Different types of sources: omnidirectional or cardioid

It depends on the wishes of the composer

Encoding side: describe the sound scene for a cell + head rotation

VirChor deals with cell listener for the encoding part

VirChor deals with the real listeners for interaction (GPS data -> virtual listener who can interact with the sounds)

Cell listener: scene point of view sent to the composition

Non frequency dependent directivity

Several directional patterns with lookup table (5 degrees grid)

Each sound source has a directivity tag (directivity ID)

Relative vector orientation of the sources wrt to the listeners

Optimization of message flow

What are the relevant audio parameters to optimize the data flow between graphics and audio? Is sound level a relevant notion?

Are there rules for the computation of scene simplification

1 cell = 1 region of influence

threshold for the distance

Purpose: to reach a maximum of 10 sound sources/cell through simplification

The volume of the sound sources could be a theoretical volume that does not change during the composition

Alternative solution: encode the 200 sources without considering any optimization

In this case make a test scene with 200 sources and 200 listeners and 64 cells

Cluster of machines: one machine for 8 cells

We will not have the capacity to stream 200 sources at a time. If they are small samples they can be loaded in memory. A solution could be to use multi-channel files

Possible optimizations

make clusters of source with one single orientation and localization

fog

angular threshold optimization

distance threshold as a function of $a\ r3 + b\ r2 + c\ R + d$

Todo list for graphics

- Make a scene for 2 grid listener and 200 sources
- Thread source position computation for grid listeners
- Update real listeners according to GPS coordinates and rotations
- Update sources according to composition messages
- Compute the relative positions and orientations of sources wrt each grid listener
- Compute the absolute position of each real listener wrt grid listeners

Monday 21th april Meeting: Graphics work for audio rendering

Indications concernant l'avancement du travail déjà réalisé par Perrine MONJAUX au sein du LIMSI-AMI

Ce document explicite les différents éléments déjà mis en place à ce jour concernant l'aspect graphique du projet SoundDelta, ainsi que certains des développements à venir. Ces développements sont réalisés sous forme de petits modules facilement exportables vers une autre scène autre que celle actuelle que l'on peut qualifier de scène test.

Compilation de VirChor sous Mac OS X

- Compilation de VirChor en ligne de commande sous Mac OS X 10.4 et 10.5
- Création de projets xcode de VirChor pour Mac OS X 10.4 et 10.5
- Création d'un package contenant tous les éléments nécessaires à la compilation de VirChor sous Mac OS X

La scène et sa décomposition cellulaire

La scène est formée d'un plan sur lequel des auditeurs réels sont visibles ainsi que des sources sonores. Pour l'instant, cette scène ne possède que deux auditeurs réels ainsi que cinq sources sonores.

L'ensemble de la communication actuelle dans la scène test se fait entre VirChor et Pd.

La scène est divisée en cellules virtuelles à l'intérieur desquelles existe un auditeur virtuel central invisible. Ces cellules ne sont pas visibles tout le temps, mais pour des besoins de développement leur affichage est permis grâce aux touches v et V du clavier. Chaque cellule possède un identifiant spécifique afin de permettre le repérage spatial.

Les sources sonores

Actuellement, les sources sonores:

- ne sont pas prédéfinies à l'avance, uniquement le nombre maximal de sources est fixé. Les sources sont ensuite ajoutées à l'aide de Pd.
- sont manipulables directement par la souris afin de les repositionner
- sont représentées par des cercles à diamètre variable, sans outil défini autre que Pd pour modifier ce diamètre.

- sont activables/desactivables mais de même que le paramètre de taille, aucun outil graphique n'est défini pour rendre manipulable cette fonctionnalité.

Les auditeurs réels

Les auditeurs réels sont représentés à l'aide d'un triangles. A l'avenir, l'utilisation d'un triangle pour cette représentation n'aura pas de sens, étant donné que l'information concernant l'orientation de la tête de l'auditeur sera gérée en local sur les terminaux mobiles.

Les auditeurs réels interagissent au sein de VirChor avec la structure cellulaire. Dès qu'un auditeur entre dans une cellule, un message est renvoyé vers Pd pour indiquer que l'auditeur ayant l'identifiant n interagit avec la cellule courante c. De plus, le message indique aussi l'identifiant de la cellule la plus proche p.

Pour l'instant, un message supplémentaire est envoyé vers Pd, sur la position absolue de l'auditeur réel dans la scène.

Prochainement, les vecteurs de position des auditeurs réels relatifs aux auditeurs n et aux cellules c et p vont être renvoyées.

Le nombre d'auditeurs réels est fixé par avance et aucun message n'est envoyé vers Pd concernant leur état (présent ou absent). Par contre pour l'instant, si un auditeur est invisible (car créé à partir de Pd) il est considéré comme inactif donc absent.

Compte rendu de la réunion REMU du 21/4/2008

Déplacement d'un objet à la souris = déplacement des objets dans leur repère

Scène complexe: utiliser la projection orthographique. Pan = translation, zoom = changement de volume de vue

Blender multi-caméras -> reportées comme des viewpoints dans VirChor, mais un seul utilisateur par fenêtre dans VirChor. Donc si les caméras ont des volumes de vue différents il faut modifier dynamiquement l'utilisateur.

Bloquer la navigation dans VirChor via les pas de translation et rotation (éviter les rotations de la scène). On peut également utiliser un mode "zoom_pan" pour la navigation, pas encore documenté car récent.

Visions subjectives: choisir par interface externe (Max, Java, Perl TK, Python TK..) -> rejeu dans Perl très précis

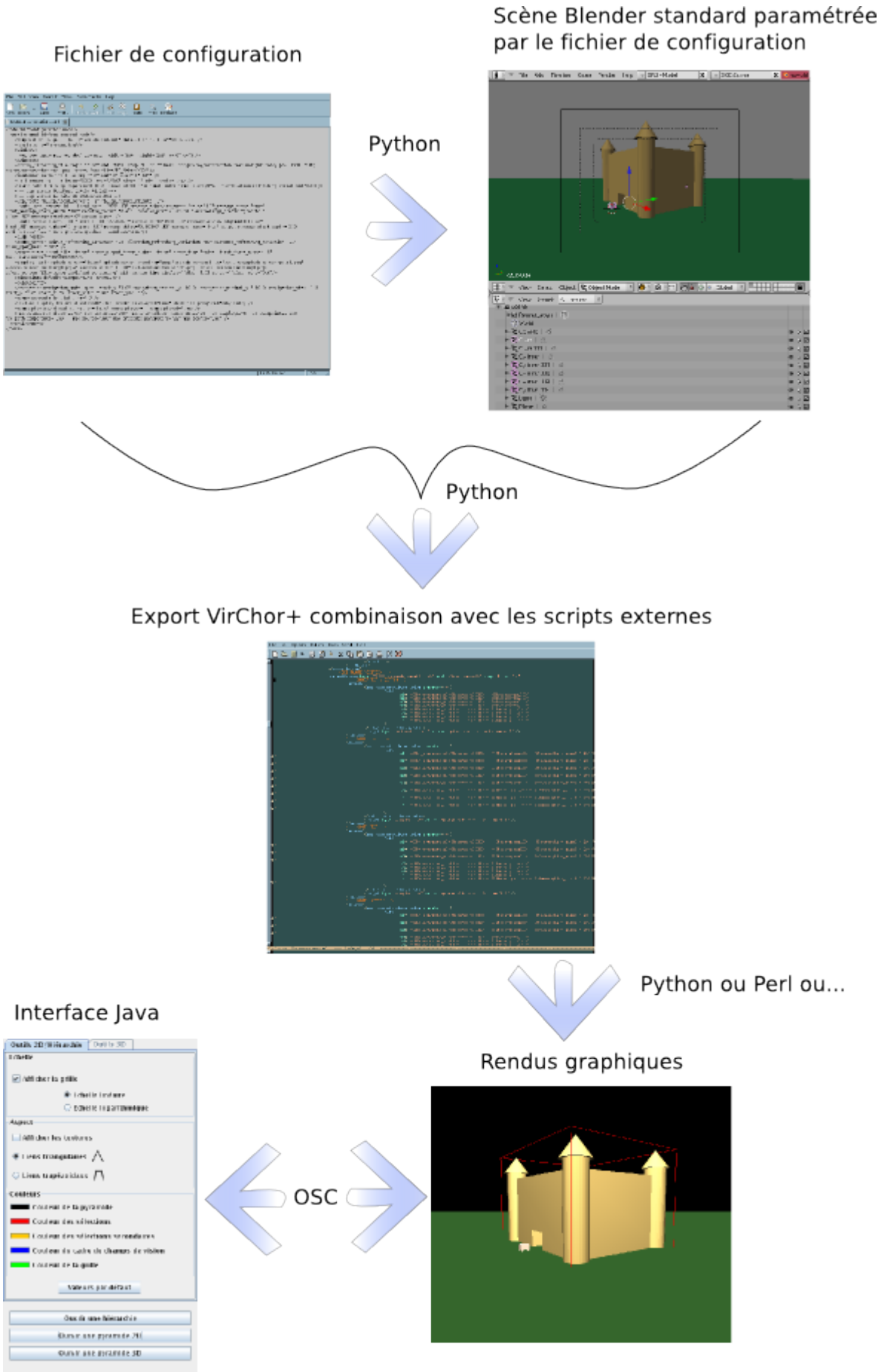
Menus dans VirChor attachés à des objets: limite des menus glut -> menus contextuels

Sources représentées par des objets circulaire + auditeurs positionnés -> faire varier fmin fmax. Avoir une petite application java compagne qui tourne en même temps

Traitement des données (r -d)/r: calcul de géométrie fait dans VirChor. Lire pour cela la taille du quad associé aux sources.

Problème de z fighting: désactiver le test de profondeur pour ces objets. Dessiner les auditeurs après les sources

---



VAE: voir une séance avec Matthieu Courgeon au LIMSI

Mettre un fichier de script assez important en externe + décoration minimale des objets + un

pogramme de fusion entre un fichier décoré et un export Blender

Avoir un document de bonne pratique pour la documentation: graphe + script + échange de messages

Rendu visuel: possibilité de travail infographique pour une visu côté spectateur. Travailler sur les shaders

Sources sonores: wave, préenregistré, synthèse, synthèse granulaire, scènes graphiques complexes

Nb de sources défini à l'avance, sources possiblement trackées. Déplacement d'une source à la souris, elles seront déplacées de façon contrôlée – fade out / déplacement / fade in

Manipulations directes: niveau, rayon, on/off, personnage tracké comme s'il est une source

Contrôle graphique du son via Max/interface java

Le son produit est traité pour être visualisé dans un outil externe

Vision subjectives + exploration de la scène. Mettre 1 caméra par spectateur

Avoir un ensemble d'identificateur permettant le broadcast

Avoir une scène Blender de base que l'on peut paramétrer par un script python + fusion avec des scripts VirChor pour générer 95% des données de la scène avec des fichiers de configuration

cellules radio / cellules audio / auditeurs virtuels / auditeurs réels

Contrôle des auditeurs simulés ou pilotés: le compositeur peut écouter tout auditeur et peut utiliser

Utiliser le multi-vues dans VirChor pour voir simultanément des vues globales et des vues locales

Sauvegarde et rejeu: futur threading -> time stamps et rejeu. Du côté VirChor: capture des évts, stockage et rejeu dans un script Perl

Forme des sources: disques, rectangles. Pour connaître la présence d'un objet dans une zone, utiliser des sensors + des volumes parallélépipédiques. Utiliser les états des objets pour paramétrer des comportements. La réponse à des messages peut dépendre de l'état d'un objet.