1 Préambule

Ce document vise à présenter les études qui ont été menées en lien avec la manipulation de données spatiales dans le cadre du projet OSSIA.

Il existe trois défis majeurs : la conception d'un modèle de données générique et assez puissant pour couvrir les cas que nous allons présenter, la représentation graphique de ces données lorsqu'un grand nombre de contrôles sont présents, et le contrôle et l'édition par l'utilisateur.

Nous analyserons plusieurs types de projets différents dans lesquels une notion de contrôle spatial existe, et qui sont utiles à la spécification des besoins.

Puis, nous présenterons les solutions et prototypes qui ont été développés dans le cadre du projet OSSIA, afin de tenter de répondre à au moins partie des besoins qui ont été exprimés.

2 Exemples

Les exemples sont répartis en trois catégories : ceux liés aux problématiques spatiales dans un contexte sonore et musical, ceux liés aux conceptions d'interfaces, et un cas de contrôle robotique.

2.1 Cadre sonore

La spatialisation en son, puis musique est une pratique très courante et une recherche importante a eu lieu sur ce sujet.

Il existe deux problèmes distincts que l'on place parfois sous le même terme de spatialisation du son :

— La composition du son avec des composantes spatiales : conception de trajectoires qui vont faire tourner le son, etc. [1]

— Les méthodes et algorithmes permettant le rendu de cet espace avec un ensemble de hauts-parleurs ou au casque (HRTF, etc.).

C'est le premier point qui nous intéresse ici. Nous pouvons le préciser encore plus en étudiant les fonctionnements et conceptions différentes qu'il peut y avoir selon que l'objectif soit de composer de la musique, ou bien de concevoir des systèmes interactifs.

2.1.1 Trajectoires sonores

Dans un cadre musical, on aura tendance à composer par pistes, et donc à vouloir spatialiser piste par piste. Des objets sonores seront disposés dans un séquenceur tel que Pro Tools ou Reaper, puis, au moment de la composition ou bien lors d'une interprétation en concert, les différentes pistes sont envoyées sur différents canaux audio.

1. *Vocabulaire de l'espace en musiques électroacoustiques*, 2006, Bertrand Merlier, et*La spatialisation des musiques électro-acoustiques*. 2012, Laurent Pottier.

Il existe des méthodes plus sophistiquées permettant de définir des trajectoires graphiquement, comme IanniX en figure1, ou bien à l'aide de modèles inspirés de lois physiques, auquel cas des applications spécifiques sont développées pour la composition. Par exemple, les modèles masse-ressort sont souvent utilisés dans ce but.

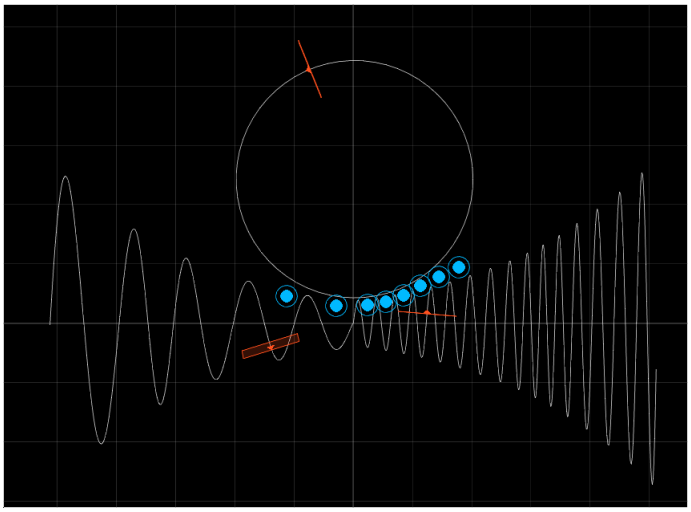

FIGURE 1 – Une partition composée de trajectoires et de triggers dans le séquenceur graphique IanniX

2.1.2 Positionnement d'objets sonores

Dans un cadre muséographique, on a plus souvent des objets sonores qui peuvent être associés à des éléments graphiques : des notes dans un manuscrit, un personnage sur un tableau... On désire alors contrôler le comportement de cet objet sonore au fil du temps et le scénariser, le faire se déplacer, changer ses propriétés sonores, etc.

Cette approche est aussi plus pertinente dans un cadre vidéo-ludique : en effet, les outils de conception de jeu-vidéo (*citer partie rapport cnam*) ont une approche fortement orientée objet - la méthode actuellement la plus répandue dans l'industrie est le pattern Entity-Component-Systems [2], qui permet d'encapsuler différentes couches applicatives (qui peuvent contenir des données, des comportements...) très simplement dans n'importe quel objet du jeu.

Voici plusieurs examples d'applications :

Application 1 : Sonopluie Ici, des utilisateurs font un parcours dans une ville. Ils ont un casque et sont géolocalisés. En fonction de leur position et de leur

2. Component Based Engine Design, Randy Gaul : http ://www.randygaul.net/2013/05/20/component-based-engine-design/

parcours, ils entendent des messages rappelant des évènements historiques de la ville.

De plus, si des utilisateurs se croisent, l'application peut en avoir conscience et déclencher des évènements spécifiques.

Cette application a plusieurs pré-requis :

— L'outil de conception doit, au minimum, permettre de placer des points sur une carte et de définir une action sur le périphérique de l'utilisateur, lorsqu'il rencontre un point.

— Pour plus de richesse, des zones sonores peuvent être définies : par exemple, une bande de plage.

— Il doit y avoir une notion de scénarisation telle qu'offerte par i-score, permet de construire une histoire au cours du temps, plutôt que de simplement avoir une liste de hot-spots interactifs.

— Enfin, se pose la question de la répartition et de l'interaction entre plusieurs protagonistes. Une possibilité serait de définir des zones d'interactions qui ne s'activent que lorsque plusieurs participants sont présents à l'intérieur. Les périphériques des utilisateurs présents dans cette zone doivent communiquer entre eux (ou bien communiquer via un serveur, mais cela nécessite des appareils possédant une connection internet).

Application 2 : Le promeneur écoutant cf. rapport CNAM

2.2 Cadre IHM

Ici, nous nous intéressons aux possibilités de création et de contrôle dynamique d'interfaces graphiques. La plupart des paradigmes d'UI permettent de définir des vues dans un éditeur graphique, mais ces vues sont en général statiques et il est nécessaire de recourir à des langages de programmation pour avoir une évolution dans le logiciel *(à vrai dire il y a des outils graphiques pour ça dans Qt par exemple, j'imagine que ça doit pouvoir se faire assez bien dans Flash aussi ?)*.

Nous aborderons ici deux cas : le premier est celui des interfaces utilisateur programmables graphiquement, ce qui peut s'avérer utile quand il y a de nombreuses animations et objets graphiques non-usuels. Le second est celui des interfaces nécessaires pour contrôler des espaces de paramètres particuliers.

Il reste des problèmes non abordés : notamment la création d'objets dynamique, la visualisation, et la gestion de comportements d'objets en masse, comme par exemple pour un générateur de particules.

2.2.1 Contrôle d'interfaces utilisateurs

Le premier cas est celui d'une application en mode kiosque, possédant un menu de choix des langues, qui mène pour chaque langue à un menu principal permettant de choisir des vidéos à lire en mosaïque.

Lorsqu'une langue est choisie, l'écran principal glisse depuis la droite. Les boutons ont une animation lorsqu'ils sont pressés. Lorsqu'une vidéo est pressée,

elle se met en lecture. Des *gestures* permettent de déplacer les vidéos, en suivant des lois de physique basiques (notamment, les vidéos peuvent tourner et rentrer en collision).

(*note : s'il y a une vraie application muséo à la place ici je suis preneur :) j'ai juste imaginé des choses pour faire un example.*)

Les problématiques, ici, sont :

— Le contrôle d'un grand nombre d'élements statiques, et la hiérarchisation de ces éléments. Par exemple, si les boutons de langues sont enfants du premier menu, on veut qu'ils suivent le déplacement de leur menu parent.

— Un questionnement sur les modèles physiques. Cela a déjà été abordé pour les modèles masse-ressort évoqués en spatialisation de musique.

— La gestion d'animations. Généralement, une animation consiste en l'interpolation de paramètres dans le temps, en passant par des états prédéfinis (keyframes).

2.2.2 Navigation dans espaces de paramètres

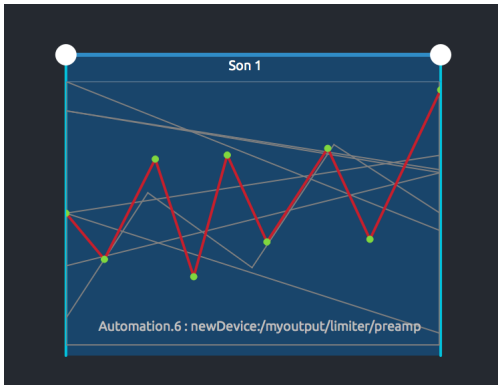

FIGURE 2 – De nombreux paramètres sur un seul axe dans i-score, rendant la partition peu lisible.

Un problème plus général est celui de la gestion des espaces de paramètres particuliers.

Par exemple, pour choisir et animer la couleur d'un élément dans le temps, il est nécessaire d'avoir des outils de création adaptés, tout en gardant les primitves de trajectoires et d'objets évoquées précédemment.

Un second point est celui de la gestion des correspondances entre espaces de données. Ainsi, si on désire automatiser un point sur un plan en deux dimensions dans le temps, il doit être possible de le faire selon les systèmes de coordonnées cartésiennes ou polaires.

De même, pour les couleurs il est important de pouvoir travailler en RVB, mais aussi en L*a*b*, HSL, et notamment de pouvoir passer facilement d'un

mode à l'autre.

Enfin, le dernier point est le cas d'une aide à l'écriture pour une application comportant un grand nombre de paramètres, potentiellement artistiques. Le problème de la visualisation se pose lorsque ces paramètres sont représentés sur une seule dimension, comme en fig. 2.

2.3 Contrôle de robots

Une application intéressante est celle des chorégraphies de robot. Ce chantier, commencé en 2015, vise à étendre les possibilités du séquenceur i-score d'abord à des robots quadrupèdes Metabots (fig. 3), puis, potentiellement à des drones.

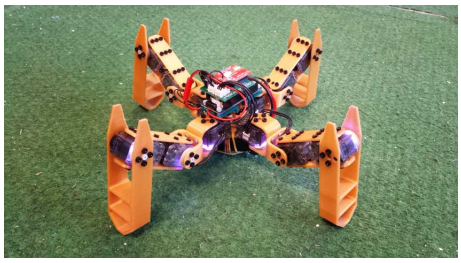

Figure 3 – Un metabot.

Là encore, la question de l'écriture se pose, et notamment le cas de la gestion d'un grand nombre d'objets dans un espace relativement restreint.

De plus, ces objets ne vont pas forcément réagir comme prévu : si par exemple un robot est programmé pour aller tout droit, mais qu'il rencontre un autre robot qui serait tombé en panne sur le chemin, actuellement le premier robot ne saurait en tenir compte et rentrerait en collision avec le second robot.

3 Réalisations

Les réalisations se séparent en deux catégories : la première consiste en des extensions pour i-score ou d'autres logiciels et frameworks. Ces logiciels ont été retenus car ils sont directement en lien avec les problématiques qui ont été évoquées précédemment.

Puis, une fois ces cas étudiés, nous avons tenté d'apporter une solution originale au problème d'écriture qui se pose.

3.1 Clients manipulant des données spatiales

3.1.1 Interopérabilité avec Qt

Un plug-in pour i-score permettant d'exposer l'arbre d'objets d'une application Qt a été développé. Cela permet de contrôler facilement la position et le

contenu des éléments d'une interface graphique réalisée avec cette technologie depuis i-score.

3.1.2 Contrôle d'OpenAL

OpenAL est une bilbiothèque ayant été développée pour les besoins des jeux vidéos au début des années 2000. Elle se base sur une notion de sources sonores associées à des fichiers sons, qui sont positionnées en trois dimensions, ainsi que d'un listener qui se déplace et perçoit donc les sons plus ou moins forts. Une implémentation récente, OpenAL-soft, permet d'utiliser des HRTF pour avoir une sensation de spatialisation réaliste au casque.

Un outil exposant des sources OpenAL via le protocole Minuit et permettant de charger des sons, puis de les positionner dans l'espace a été implémenté. Ces sources sont donc contrôlables depuis i-score.

3.1.3 Contrôle de metabots

Une couche de communication entre i-score et les robots Metabots a été développée et a mené à une présentation aux Robot Makers Day, avec notamment une chorégraphie ou deux robots sont contrôlés et synchronisés musicalement avec i-score.

Cela a été l'occasion de comparer le développement d'une chorégraphie entre i-score, et un langage de programmation graphique inspiré de Scratch. Les résultats ont été probants : il a fallu beaucoup moins de temps pour produire une même chorégraphie dans i-score que dans l'autre langage.

3.2 État de l'art sur méthodes spatiales, positionnement, trajectoires

3.2.1 Free-hand drawing sur tableau

À titre d'exemple pour des démonstrations et exposés, un outil permettant de dessiner des zones de manières arbitraire sur un tableau a été développé (cf. fig. 4).

Lorsqu'une zone est cliquée, elle produit un son.

3.2.2 Liste des tâches et prototype pour processus de mapping spatial

Suite aux différents cas d'utilisation qui ont été présentés, un état de l'art a été effectué dans le domaine de l'authoring spatial, et des techniques et méthodes possibles.

4 Conclusion

Problèmes restants : question de visualisation / rendu, etc.

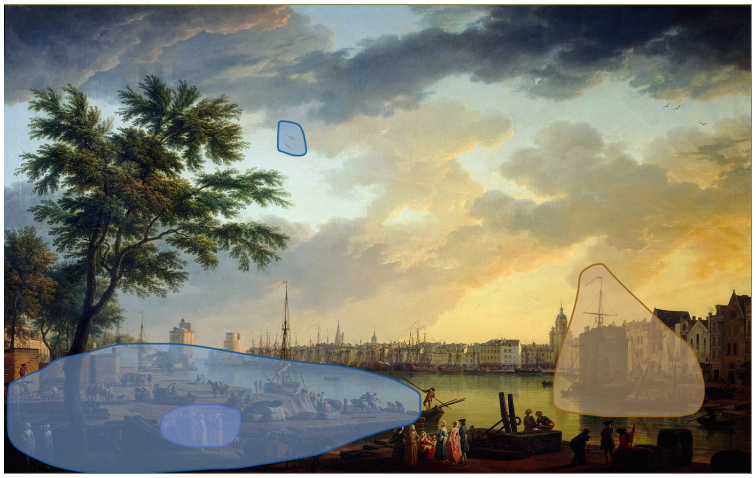

Figure 4 – Un tableau sur lequel on peut apposer des couches sonores.