Air Instruments

Spécifications des requis du système (SRS)

Version 1.1

Historique des révisions

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Auteur** |
| 2014-01-15 | 1.0 | Ébauche du SRS | Alexandre Vanier |
| 2014-01-22 | 1.1 | Amélioration du SRS et ajouts de requis | Alexandre Vanier |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table des matières

1. Introduction 5

1.1 But 5

1.2 Définitions, acronymes et abréviations 5

1.3 Vue d’ensemble du document 5

2. Description globale 5

2.1 Caractéristiques des usagers 5

2.2 Interfaces 5

2.2.1 Interfaces usagers 5

2.2.2 Interfaces matérielles 5

2.2.3 Interfaces logicielles 5

2.2.4 Interfaces de communication 6

2.3 Contraintes générales 6

2.4 Hypothèses et dépendances 6

3. Exigences fonctionnelles 6

3.1 Instruments obligatoires 6

3.2 Exigences relatives au piano 6

3.2.1 Vue à la première personne 6

3.2.2 Positionnement du capteur 6

3.2.3 Notes jouables 6

3.2.4 Grosseur relative des modèles 7

3.2.5 Utilisation des pédales 7

3.3 Exigences relatives aux instruments à percussion 7

3.3.1 Vue à la première personne 7

3.3.2 Jouabilité de la batterie 7

3.3.3 Positionnement du capteur 7

3.3.4 Pédale de la batterie 7

3.4 Instruments optionnels 7

3.5 Mode libre 7

3.6 Mode assisté 8

3.7 Mode néophyte 8

3.8 Détection automatique des instruments 8

3.9 Affichage à la première personne 8

3.10 Affichage d’une scène 8

3.11 Nombre de joueurs 8

3.12 Deux pianos en simultané 8

4. Exigences non-fonctionnelles 9

4.1 Utilisabilité 9

4.1.1 Temps de formation 9

4.1.2 Temps de délai du son 9

4.1.3 Temps de délai de l’image 9

4.2 Fiabilité 9

4.2.1 Temps moyen entre pannes 9

4.2.2 Temps moyen jusqu’à la réparation 9

4.3 Performance 10

4.3.1 Utilisation de ressource 10

4.4 Maintenabilité 10

4.4.1 Calibrage facile 10

4.4.2 Normes de codage 10

4.5 Contraintes de conception 10

4.5.1 Langage de programmation 10

4.5.2 Librairies utilisées 10

4.5.3 Processus 10

4.5.4 Achat de composantes 11

4.6 Exigences de la documentation usager en ligne et du système d’assistance 11

ANNEXE A: Glossaire 12

Spécifications des requis du système (SRS)

# Introduction

## But

Le SRS décrit le comportement externe d’une application. Il décrit aussi les exigences non fonctionnelles, les contraintes de conception, ainsi que les autres facteurs nécessaires à la description complète des exigences du logiciel à développer.

## Définitions, acronymes et abréviations

## Vue d’ensemble du document

Le document ci-dessous présente tout d’abord une description globale du système puis présente ensuite les exigences fonctionnelles organisées selon certains thèmes (ici, par instrument). Il y a finalement les exigences non-fonctionnelles qui sont organisées par des thèmes prédéfinis.

# Description globale

Air Instruments est un logiciel à but récréatif permettant à l’utilisateur de jouer de la musique sans avoir besoin d’instruments physiques. Le logiciel détecte les mouvements faits par l’utilisateur à l’aide d’un capteur et effectue un traitement qui permet d’obtenir un son réaliste.

## Caractéristiques des usagers

Air Instruments est un projet réalisé dans le cadre de la compétition Laval Virtual, il est donc adapté au grand public mais en visant plus spécifiquement les experts en réalité virtuelle qui jugeront le produit. Autrement dit, le logiciel doit être performant du point de vue technologique mais tout de même simple à utiliser pour un utilisateur lambda.

## Interfaces

### Interfaces usagers

L’interface du jeu sera minimale : l’écran indiquera dans tous les modes une aide montrant quels gestes effectuer pour chaque instrument ainsi qu’une vue modélisée dépendant de l’instrument joué. Avant que le jeu débute, un menu très simple indiquera les modes sélectionnables ainsi qu’une brève description de chaque mode.

### Interfaces matérielles

Le logiciel utilisera un ou plusieurs capteurs de mouvement tel Kinect ainsi qu’un projecteur pour projeter les images traitées sur une toile. Il est aussi possible que des écrans supplémentaires soient utilisés pour séparer l’interface du joueur de ce qui est vu par les spectateurs. Il faudra aussi des haut-parleurs pour faire jouer le son des instruments.

### Interfaces logicielles

Le moteur Unity sera utilisé pour la partie qui sera affichée à l’écran (les modèles 3D et leur logique d’affichage). Un pilote et probablement un SDK seront utilisés pour gérer la capture d’images et pour avoir des fonctions de traitement de base. Nous utiliserons aussi les librairies OpenCV et OpenCL afin de faire du traitement d’images et de faire des calculs en parallèle sur le GPU.

### Interfaces de communication

La/les Kinect (ou autre) sera branchée à un ordinateur qui recevra les images et les traitera. La scène à afficher sera ensuite projetée à l’aide d’un projecteur et la vue en FPS pourra possiblement être transmise à des écrans supplémentaires installés en face du/des joueurs.

## Contraintes générales

L’espace disponible en compétition sera de 2m x 2m. Le logiciel doit fonctionner bien dans une pièce de ces dimensions. Le logiciel doit aussi fonctionner avec un niveau de bruit élevé et avec un éclairage variant.

## Hypothèses et dépendances

Nous émettons l’hypothèse que la Kinect calculera de façon précise et rapide les joints et fournira des images que nous pourrons traiter.

# Exigences fonctionnelles

## Instruments obligatoires

Le piano et la batterie devront être jouables.

**Essentiel**

## Exigences relatives au piano

### Vue à la première personne

La vue à la première personne pour le piano sera une représentation comprenant les mains modélisées du joueur avec un piano modélisé en dessous. La caméra (Unity) devra être placée de façon à ce que le joueur puisse bien voir la hauteur de ses mains par rapport au piano afin de faciliter l’utilisation de celui-ci.

**Essentiel**

### Positionnement du capteur

Le capteur devra être placé au sol de façon à ce que celui-ci puisse détecter plus facilement la profondeur des mains.

**Essentiel**

### Notes jouables

Toutes les notes d’un piano classique devront être jouables, incluant les noires et les blanches.

**Essentiel**

### Grosseur relative des modèles

Le modèle du piano devra contenir des notes qui sont d’une taille un peu plus grande que la largeur des doigts afin de faciliter un fonctionnement précis du capteur et du traitement d’images.

**Essentiel**

### Utilisation des pédales

Il sera possible d’utiliser les pédales du piano en plaçant les pieds au sol à un endroit défini.

**Optionnel**

## Exigences relatives aux instruments à percussion

### Vue à la première personne

Une vue à la première personne sera affichée lorsque le capteur détectera que l’utilisateur effectue un mouvement semblable à un joueur de batterie (mouvement des bras). La vue présentera une vue des baguettes qui s’agitent sur un modèle de batterie.

**Essentiel**

### Jouabilité de la batterie

Tous les instruments à percussion dans une batterie seront jouables et auront un fonctionnement similaire.

**Essentiel**

### Positionnement du capteur

Le capteur pour la batterie sera positionné en face et détectera les mouvements des bras afin de savoir quels instruments sont joués à l’instant.

**Essentiel**

### Pédale de la batterie

La pédale de la batterie sera jouable.

**Essentiel**

## Instruments optionnels

La guitare, divers instruments à percussions et divers instruments à vent pourront être jouables.

**Souhaitable**

## Mode libre

Un mode libre sera disponible où l’utilisateur pourra jouer de l’instrument choisi. Dans ce mode, seul le son des instruments sera joué par les haut-parleurs.

**Essentiel**

## Mode assisté

Un mode assisté sera disponible. Dans ce mode, une musique présélectionnée sera jouée. Lorsque l’utilisateur jouera d’un instrument, cet instrument sera enlevé de la musique jouée et seulement les autres instruments seront joués. Lorsque l’utilisateur arrête de jouer, l’instrument sélectionné est rajouté à la musique automatiquement.

**Essentiel**

## Mode néophyte

Un mode néophyte sera disponible. Dans ce mode, l’utilisateur sera assisté par le logiciel dans la pratique de l’instrument choisi. Pour un piano, par exemple, les notes à jouer pourraient être mis en surbrillance à l’avance ou alors on pourrait fournir l’enchainement de notes à suivre dans une sorte de « tableau » à la Guitar Hero/Rock Band.

**Optionnel**

## Détection automatique des instruments

Les instruments devront être détectés automatiquement. Ainsi, lorsqu’un joueur effectue un mouvement similaire à celui de quelqu’un qui jouerait du piano, un piano apparaîtra à l’endroit où il joue afin de servir de référence. Un mécanisme similaire sera utilisé pour tous les instruments.

**Essentiel**

## Affichage à la première personne

Un affichage à la première personne sera affiché afin d’aider l’utilisateur à percevoir sa position par rapport à l’instrument. Cette vue dépendra de l’instrument détecté au départ et pourra changer si l’utilisateur change d’instrument en cours de route.

**Essentielle**

## Affichage d’une scène

Une scène sera affichée où chaque joueur sera montré avec l’instrument joué près de lui (comme s’il jouait d’un vrai instrument). Un décor pourra être affiché en arrière.

**Optionnel**

## Nombre de joueurs

Le nombre de joueur maximal sera de 2.

**Essentiel**

## Deux pianos en simultané

Il sera possible pour les deux joueurs de jouer du piano de façon simultanée.

**Optionnel**

# Exigences non-fonctionnelles

## Utilisabilité

### Temps de formation

Un utilisateur normal doit pouvoir utiliser le logiciel facilement, avec une période d’apprentissage maximum de quelques minutes.

### Temps de délai du son

Le son de chaque note pour chaque instrument devra être joué avec un délai maximal de quelques millisecondes après que l’action soit effectuée par l’utilisateur.

### Temps de délai de l’image

La scène sera affichée avec un délai maximal de quelques de dizaines de millisecondes après que l’action soit effectuée par l’utilisateur.

## Fiabilité

### Temps moyen entre pannes

Le temps moyen entre panne devra être très grand, avec un 1 panne par 9 heures.

### Temps moyen jusqu’à la réparation

Le système devra être facilement réparable et/ou calibrable, avec un temps moyen de réparation d’au maximum quelques minutes et un temps de démarrage d’au maximum deux minutes.

## Performance

### Utilisation de ressource

Le système devra tenter d’utiliser le maximum de ressources dans la mesure où cela permet d’améliorer les délais de traitement (autrement dit, les délais de traitement seront privilégiés). Il faut cependant s’assurer que la charge de traitement peut être supportée par des ordinateurs portables.

## Maintenabilité

### Calibrage facile

Le système doit être facile à calibrer afin que le système puisse être réutilisé par le département de génie logiciel pour maximiser le rayonnement de l’École Polytechnique.

### Normes de codage

Les conventions de codage utilisées seront le google style guide pour C++ et les conventions de Microsoft pour le C#.

## Contraintes de conception

[Cette sous-section doit indiquer toute contrainte de conception au système développé, comme le langage de programmation, le processus logiciel, l’achat de composantes, les libraires de classes, etc.]

### Langage de programmation

Le langage utilisé sera le C# et le C++.

### Librairies utilisées

Le moteur 3D Unity sera utilisé ainsi que diverses librairies en lien avec le capteur et/ou avec la transmission réseau. Les libraires OpenCV et OpenCL seront aussi utilisées pour le traitement d’images et les calculs en parallèles sur GPU ainsi que la librairie de Base de google pour le C++.

### Processus

Le processus utilisé sera une variation des processus agiles. Ce processus est défini plus en détails dans un schéma.

### Achat de composantes

Il faudra se procurer un capteur (ex une Kinect) et possiblement divers matériaux afin d’avoir un bon montage pour maximiser la performance des capteurs.

## Exigences de la documentation usager en ligne et du système d’assistance

Il n’y aura pas de documentation usager excepté une aide en jeu indiquant les mouvements nécessaires pour simuler chaque instrument.

# ANNEXE A: Glossaire

[Si vous n’utilisez pas cette annexe, veuillez la supprimer complètement.]

|  |  |
| --- | --- |
| **Terme** | **Description** |
| SDK | *Software Development Kit* |
| FPS | Vue à la première personne (*First person shooter*) |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |