|  |
| --- |
|  |
| Contrôle d’avatar par capteur Kinect |
| *Prenez le contrôle !* |
|  |
| **BARBESANGE Benjamin & GARÇON Benoît** |
| **juin 2015** |

|  |
| --- |
|  |

Remerciements

Nous tenons à remercier Monsieur Emmanuel Mesnard qui nous a permis de réaliser un projet qui nous tenait à cœur mais aussi pour l’aide qu’il nous a fourni tout au long de ce projet.

Nous voudrions aussi remercier notre ami Simon Leschiera pour nous avoir prêter du matériel et nous avoir permis d’avancer plus rapidement dans notre projet.

Table des matières

[Remerciements 1](#_Toc421566658)

[Présentation du projet 3](#_Toc421566659)

[Origine du projet 3](#_Toc421566660)

[Etude préliminaire du projet 3](#_Toc421566661)

[Etude du matériel 4](#_Toc421566662)

[Processing 4](#_Toc421566663)

[Capteur Microsoft Kinect 2 4](#_Toc421566664)

[Capteur Microsoft Kinect 1 4](#_Toc421566665)

[Travail sur les données 5](#_Toc421566666)

[Identification des données 3D à traiter 5](#_Toc421566667)

[Formats et contraintes existantes 5](#_Toc421566668)

[Solution retenue : un format personnel 5](#_Toc421566669)

[Prise de contrôle et optimisations 6](#_Toc421566670)

[Algorithme d’animation 6](#_Toc421566671)

[Principe 6](#_Toc421566672)

[Application 6](#_Toc421566673)

[Améliorations testées 6](#_Toc421566674)

[Lissage 6](#_Toc421566675)

[RAZ : remise à zéro 6](#_Toc421566676)

[Calcul du mouvement indépendant du modèle 6](#_Toc421566677)

[Conclusion 7](#_Toc421566678)

[Bilan du projet 7](#_Toc421566679)

[Améliorations possibles 7](#_Toc421566680)

[Annexes 8](#_Toc421566681)

[Création d’un avatar 8](#_Toc421566682)

[Utilisation de Processing sous Eclipse 9](#_Toc421566683)

Présentation du projet

# Origine du projet

Dans le cadre de notre projet de première année à l’ISIMA, nous avons choisi de développer un programme de contrôle d’avatar 3D par une Kinect. Ce projet s’inscrit dans la continuité du cours de réalité virtuelle de deuxième année de Prep’ISIMA.

L’objectif sera donc de réussir à animer à l’écran un personnage (humanoïde) grâce aux mouvements de l’utilisateur numérisés par le capteur Kinect. Il s’agira donc dans un premier temps de développer un programme sous Processing capable de mettre en œuvre un tel mécanisme.

# Etude préliminaire du projet

Ce projet est beaucoup plus conséquent qu’il ne pouvait le laisser paraître. En effet, il faut tout d’abord s’approprier le matériel : des capteurs propriétaires avec des compatibilités parfois limitées avec certaines technologies. Il convient donc d’étudier toutes les possibilités offertes par le matériel.

Ensuite afin d’avoir une compatibilité reconnue avec les technologies informatiques nous avons décidé d’utiliser la bibliothèque Processing, en effet cette bibliothèque nous est familière et c’est un bon compromis entre haut niveau de programmation et programmation précise. Il existe aussi Processing sous forme d’environnement de développement toutefois pour des raisons de confort de développement il a été décidé d’utiliser Eclipse afin d’avoir accès aux différents outils qu’il propose. Afin d’utiliser Processing correctement dans Eclipse il nous a été nécessaire d’installer le plugin Proclipsing, cette démarche est expliquée plus en détail en annexe.

Une fois tout ceci en place il nous a fallu ajouter les bibliothèques annexes pour l’utilisation du matériel à savoir SimpleOpenNi et KinectPV2. Les problèmes ont été nombreux avant de pouvoir compiler un exécutable. Mais après avoir obtenu une version patchée des bibliothèques, la compilation et l’exécution du programme était possible.

Alors il nous a fallu rechercher le meilleur moyen de représenter des personnages virtuels et ensuite de les animées, grâce à une animation squelettale. La modélisation des données a été dans ce projet la partie majeure et a demandé beaucoup de recherches.

Enfin, quand notre objectif initial a été atteint, nous nous sommes appliqués à mettre en place des optimisations et améliorations de notre premier algorithme d’animation naïf.

# Présentation du programme

Le programme principal contenu dans le fichier ZZavatar.pde/ZZavatar.java est un programme mono-utilisateur repérant l’utilisateur à l’aide d’un capteur Kinect (version 1 ou 2 indifféremment). Les mouvements de cet utilisateur sont retranscrits à l’écran au travers d’une modélisation 3D d’un personnage humanoïde (ou non) grâce à notre algorithme de contrôle.

Un contrôle plus classique du programme est possible :

* Les touches directionnelles, ‘+’ et ‘-‘ permettent de contrôler une caméra orientées sur le personnage ;
* La touche ‘s’, ou suivant, permet de passer au modèle suivant ;
* La touche ‘d’ permet d’activer un mode debug ;
* La touche ‘f’ permet de changer le fond de la scène ;
* La touche ‘g’ permet d’activer/désactiver le fond.

La gestion des Kinects est gérée par les classes ZZkinectV1 et ZZkinectV2 qui sont toutes les deux soumises à l’interface ZZkinect ce qui nous a permis d’utiliser indifféremment l’une ou l’autre dans le code principal.

La gestion du modèle 3D est faite par la classe ZZModel, celle du squelette par ZZkeleton et les textures par ZZMaterial. Ces trois classes sont une partie majeure du traitement des données et du projet, tout le projet repose sur cette modélisation.

La classe ZZbackground gère quant à elle l’utilisation du fond d’écran.

Les classes ZZector, ZZertex et ZZoint sont des versions plus ou moins spécialisées de la classe PVector de Processing. Chacune de ces classes possèdent des méthodes et attributs qui facilitent une partie du traitement :

* ZZector est une classe mère de ZZoint et ZZertex ;
* ZZoint est plutôt utilisé pour la gestion des joints du squelette des différents personnages/utilisateurs ;
* ZZertex (venant de vertex) est une spécialisation utile pour la gestion des sommets de maillages (modèles 3D).

Nous avons aussi une classe ZZfifo qui est une simple classe

Etude du matériel

# Processing

# Capteur Microsoft Kinect 2

Microsoft Kinect 2 : Ce capteur nous permettra l’acquisition des mouvements de l’utilisateur. En effet, grâce à ce dernier nous avons accès à pas moins de 25 points représentants des zones clefs du squelette à 60fps.

# Capteur Microsoft Kinect 1

Travail sur les données

# Identification des données 3D à traiter

# Formats et contraintes existantes

# Solution retenue : un format personnel

Prise de contrôle et optimisations

# Algorithme d’animation

## Principe

## Application

# Améliorations testées

## Lissage

## RAZ : remise à zéro

## Calcul du mouvement indépendant du modèle

Conclusion

# Bilan du projet

# Améliorations possibles

Annexes

# Création d’un avatar

Nous allons à présent vous donner les indications les plus précises possibles pour créer vous-même vos propres avatars à animer :

##### ETAPE 1

La première étape consiste tout simplement à choisir un modèle ou mesh qui vous plaît et l'importer sous Blender.

##### ETAPE 2

Ensuite il convient de lui assigner l'armature appropriée avec les bons noms (il est recommandé d'utiliser ./SkeletonTemplate.blend).

##### ETAPE 3

Veillez à ce que l’armature corresponde bien à l’ossature du modèle, ensuite sélectionnez le mesh puis l'armature.

##### ETAPE 4

Nous allons maintenant lier le chaque face du maillage à un os du squelette en faisant comme suit : Ctrl+P > Set Parent to > Armature Deform > With Automatic Weights.

##### ETAPE 5

Une fois ceci fait on peut exporter les données résultantes :

* On exporte le nouveau maillage : File > Export > Wavefront (.obj) avec les options : "Include UVs" "Write Materials" "Triangulate Faces" "Polygroups" "Objects as OBJ Objects".
* Ensuite on exporte le squelette dans le format bvh : File > Export > Motion Capture (.bvh).

##### ETAPE 6

Dans Misfit Model 3D, chargez le mesh et vérifiez (et corrigez) l'assignement automatique des faces aux groupes. Pour ce faire aller dans le menu "Groups > Edit Groups...". En particulier vérifiez "No group" et d'éventuels "(null)".

##### ETAPE 7

Exportez le résultat obtenu depuis Misfit (ne pas sauver les normales).

##### ETAPE 8

Copiez tous les fichiers dans le répertoire data du programme.

Attention toutefois à veiller à bien donner le même nom à tous les fichiers.

# Utilisation de Processing sous Eclipse