|  |  |
| --- | --- |
|  | A2010 |
|  | Adrien SERIN  Henri PAYNO  Coralie VESSIERES |

|  |
| --- |
| **[**Rapport bras haptique**]** |
| Création d’un jeu de type JENGA |

Sommaire

[Introduction 3](#_Toc281939920)

[I. Création d’une DLL 3](#_Toc281939921)

[1. Introduction 3](#_Toc281939922)

[2. Résultat de la DLL 3](#_Toc281939923)

[II. Environnement du jeu 3](#_Toc281939924)

[1. Modélisation 4](#_Toc281939925)

[2. Interface 4](#_Toc281939926)

[3. Physique 4](#_Toc281939927)

[III. Gameplay 4](#_Toc281939928)

[1. Mode caméra 4](#_Toc281939929)

[2. Mode prendre 4](#_Toc281939930)

[IV. Problèmes rencontrés 5](#_Toc281939931)

[Conclusion 5](#_Toc281939932)

# Introduction

Dans le cadre de l'UV VI50 nous avons fait un projet sur le thème du bras haptique. Pour au mieux utiliser toutes les possibilités de ce bras nous avons modélisé un jeu de JENGA. Le but est de créer une interaction sensori-motrice entre l'homme et la machine pour obtenir une immersion à la fois visuelle et tactile.

Le principe du jeu est de déplacer à l'aide du bras haptique les blocs d'une tour sans faire tomber celle ci. Un retour d'effort est fourni par le bras lors du déplacement de ces blocs.

Cette idée est inspirée du jeu Boom Blox sortit sur Wii.

# Création d’une DLL

## Introduction

Afin d'interagir avec le bras haptique dans le logiciel Unity, nous avions besoin d'un DLL. Celle ci a été créée à l'aide de visual studio et des drivers fournis avec le bras haptique. La DLL nous donne la possibilité de récupérer les déplacements avec le bras mais aussi de pouvoir lui appliquer diverses forces.

Le code de cette DLL est inspiré des exemples fournis avec les drivers.

Afin de faire fonctionner le bras haptique le développement d'une dll est nécessaire. Pour cela on se base sur les modèles architecturaux proposés par Sensable.

La dll est développée en C++ et contient les fonctionnalités de bases :

* position du bras, en absolut et en relatif (différence depuis la dernière position).
* Affecter des forces au bras haptique (en x, y et z).
* Obtenir la valeur des boutons.

Afin d'utiliser la dll il faut initialiser le bras haptique (via la fonction init() de la dll) . Cette fonction lance une routine de mise à jour des attributs du bras (sur chaque frame) ainsi qu'un scheduler associé au bras haptique. Pour obtenir la valeur ds attributs il suffit de faire appel aux getters proposé par la dll.

Lorsque l'on quitte l'application il faut libérer le bras haptique via « exitHD() ». Cette fonction termine le scheduler.

## Résultat de la DLL

La dll répond aux attentes et est assez stable. Elle peut facilement être réutilisable puisque les fonctions crées sont assez génériques. L'importation des DLL sous Unity ne se fait uniquement sous la version pro. Il suffit ensuite d'ajouter un import dans un script C# pour utiliser une fonction créée dans cette DLL.

# Environnement du jeu

Le jeu est composé de plusieurs scènes jouables contenant une tour de JENGA chacun. Comme présenté dans le cahier des charges, le décor est non réaliste.

## Modélisation

Tous les terrains sont modélisés sous Unity, de même que les effets d'eau et que la skymap. Tout autre objet ou animation sont créés sous 3DS max et importés sous Unity. Les animations sont donc des scripts qui tournent en boucle et indépendamment de Unity.

Chaque tour est composée de deux parties obligatoires. Un socle qui permet de surélever la tour pour mieux se déplacer autour, et d'un poids qui représente le toit de la tour.

Les éléments animés ont été modélisés sous 3DSMax. On répertorie les caractères suivants :

|  |  |
| --- | --- |
| bogo.bmp | Nom : **Bogo**  Famille : **Gorille**  Caractéristiques : **Gros bras pas content**  Mission : **Surveiller les poulettes** |

|  |  |
| --- | --- |
| cocotte.bmp | Nom : **Cocotzilla**  Famille : **Poule**  Caractéristiques : **Maniaque du nettoyage**  Mission : **Prier pour la destruction de la tour** |

|  |  |
| --- | --- |
| toto.bmp | Nom : **Toto**  Famille : **Indéterminée**  Caractéristiques : **Confectionne des choses inutiles**  Mission : **Fuir de l’ile** |

|  |  |
| --- | --- |
| laMort.bmp | Nom : **La mort**  Famille : **N’existe pas**  Caractéristiques : **Plante des choux**  Mission : **Détruire toto** |

## Interface

Une interface de compteur de points a été créée pour le jeu. Ce compteur change lorsque qu'un bloc touche le sol. Si ce bloc est celui qu'on vient d'enlever on a une incrémentation du compteur, si au contraire le bloc tombe de lui même de la tour il décrémente le compteur.

## Physique

Pour implémenter la physique sous Unity on utilise les options Physx du logiciel. Pour cela chaque bloc de la tour, le poids compris, possède un rigidbody. Ce rigidbody est sensible à la gravité. Chaque bloc a des propriétés physique tel que sa texture (bois, métal, plastique, glace), sa masse et un coefficient de réaction au frottement.

# Gameplay

## Mode caméra

Avec le bras haptique on peut déplacer la caméra pour tourner autour de la tour pour avoir un meilleur point de vu. Ce mode est activé en appuyant sur le bouton gris clair du bras haptique.

Si aucun bouton n'est pressé on est en mode premier plan, c'est à dire que la caméra se déplace uniquement dans le viewport.

## Mode prendre

Lorsqu'on appuie sur le bouton gris foncé on passe en mode prendre, il faut alors qu'avant d'appuyer sur le bouton notre curseur se trouve en face d'un bloc. Quand le bouton est pressé c'est le bloc en face du pointeur qui est prit. Celui ci suit alors les mouvements fait avec le bras haptique. Lorsque le bouton est relâché le bloc n'est plus sous le contrôle du bras.

Chaque bloc qui peut être pris possède le script de draganddrop, le script permet d'appliquer des transformations au bloc par rapport à un périphérique extérieur (souris, bras haptique).

Lorsque l'utilisateur se saisit d'un bloc une force est ajoutée au bras haptique (dépendante de la masse du bloc ???)

# Problèmes rencontrés

L’un des problèmes majeur est la synchronisation entre le bras haptique et Unity. Ce problème apparait aléatoirement et rend le bras inutilisable ce qui amène au redémarrage de l’application.

# Conclusion